Cranfield University
School of Mechanical Engineering
Applied Mathematics and Computing

Ph. D. Thesis

Academic Year 1998-99

Zaixiang Gao

The Numerical Treatment of Curved Boundary Surfaces in an Unfitted Grid formulation for Computational Fluid Dynamics

Supervisor: Prof. C. P. Thompson

November 1999

The present thesis is submitted
in partial fulfilment of the requirements for
the degree of Doctor of Philosophy

To My Parents And My Family

Acknowledgements

I would like to thank particularly my supervisor, Prof.. C. Thompson both for the guidance and encouragement I received during this Project, and for the many stimulating discussions which have facilitated my task. I am also indebted to Cranfield University for the grant I received during this project and all staff and students within the Numerical Modelling group in Applied Mathematics and Computing, Department of Fluid Engineering and Instrumentation, in particular Dr. P. A. Lezeau and Dr. C. Guardino for their help and discussions.

I am, however, most grateful to my wife, Xiao Rong Su and our daughter Jenny J. Gao for their constant support, particularly during the more difficult time. Without their support, I could not have successfully finished this project.

Abstract

In this thesis a new Cartesian cut-cell scheme has been presented which solves the incompressible Navier-Stokes equations using a finite volume approach and a staggered grid. The equations have been solved in conservation-law form upon a network of cell which were created using a Cartesian, cell-based grid generation procedure. This method used a recursive subdivision of fixed aspect ratio (Cartesian) cells creating arbitrarily shaped polygons when the Cartesian cell straddles a boundary, using a modified polygon clipping algorithm.

The numerical scheme adopted is the finite volume formulation, adaptive multigrid algorithm, and the Cartesian cut cell method. The scheme is modified to allow accurate calculation of flow variables at the intersection of curved body boundaries on non-aligned grids. A novel method to distinguish boundary cells is developed in this project.

The outline of the thesis is as follows: firstly, the modelling and simulation of unfitted Cartesian grids, structured grids, and unstructured grids are reviewed, together with the different numerical techniques implemented in the solver. Finite volume discrete equations are then derived on structured, staggered grid. Next, having specified the solution algorithm, we introduce the Cartesian cut cell method and consider the accuracy of the solver. Results from several test cases of varying complexity are compared with those of a widely used commercial CFD package and good agreement is obtained.

The results of this method show that the Cartesian cut cell approach is second-order accurate for the two-dimensional flows. The question of performance of the algorithm is also addressed in detail, both in terms of robustness and speed of convergence by modified transformations. Good accelerations are obtained using the adaptive multigrid method but the convergence rates are often not grid-independent. Convergence rates are much faster than those achieved by single grid solvers and commercial codes, although the solver is not fully optimal. It is confirmed that multigrid methods offer a good framework for the implementation of grid adaptation, particularly in areas where the flow variable change abruptly/rapidly. Considerable gains in speed and memory usage, by one further order of magnitude, are achieved.

Contents

List of Figures

List of Tables

Notation

A	Area, Constant
a	Acceleration
b	Channel width
c	Constant
c_p	Gas Constant
e^h	Error vector
F	Force, Blasius function
f	Function, force
g	Gravity
H	Heat flux
h	distance or height
i, j, k	Unit vector in X, Y, Z direction
I	Identity tensor
I_k^{k-1}	Restriction operator for the residual
I_{k-1}^k	Prolongation operator for the correction to the solution
J	Jacobian matrix
L	Length
k	Finest grid in a multigrid computation
P	Pressure scale [Pa]
P_e	Peclet number
Q	Fluid flux [m^2/s]
R	Universal gas constant
Re	Reynolds number
r	radius
r^h	Residual vector
S	Surface area
t	time
T	Temperature
U	Velocity vector

U_0	Specified vector
u, v, w	Velocity components in x-, y-, z-directions
u^h	Exact solution of a discrete operator
$\tilde{u}^h$	Approximate solution of a discrete operator
V	Volume
W	Work
x, y, z	Cartesian coordinates

Greek symbols

α	Angle
β	Angle
ζ	Non-dimensional length
η	A single dimensionless variable
μ	Viscosity
ν	Kinematic viscosity
ξ	Non-dimensional length
ρ	Viscosity
σ	thick of the boundary layer
τ	Shear stress
τ^k	Truncation error on the level k grid
τ_w	wall shear stress
$\mathcal{E}$	Enthalpy [J/kg]
Φ	Discrete unknowns for the local solver

Chapter 1

Introduction

1.1 Scope of The Thesis

In this thesis a numerical algorithm is presented which solves the incompressible flow on a computational domain that is created using a Cartesian cell-based grid and a Cartesian cut-cell grid generation procedure. This grid generation process creates non-overlapping volumes which fill the domain whereupon the steady, incompressible Navier-Stokes equations are solved in discrete conservation law form using an upwinded, finite-volume approach. A non-traditional means of storing the data associated with the grids is used that is based upon a tree data structure, easily allowing solution-adaptive mesh refinement. This chapter addresses the implementation of grid generation schemes and discusses their advantages and disadvantages. We discuss four grid generation methods: structured grid, unstructured grid, Cartesian cell-based grid, and the Cartesian cut cell grid generations. Chapter 2 gives the literature review. We talk about the development of multigrid methods history, the benefit of using multigrid methods, the development of the Cartesian cell-based approach, and the recent progress of Cartesian cut-cell method.

In Chapter 3, the governing equation and their discretisation are analyzed. Both the governing partial differential equations and the discrete algebraic equation are solved by PAMG (Parallel Adaptive MultiGrid) and CC-PAMG (Cartesian Cut-cell

Parallel Algorithm PAMG) which are derived using finite volume method and hybrid schemes. Important features of the discretisation, the use of staggered finite volume grids and hybrid schemes are discussed and the discretisation process is described in detail. To illustrate this approach, the local truncation error is investigated by using the Taylor series expansing, a fully second-order numerical scheme is presented. The transformation of PDE is also described which is used in CFX 4.2 to compare with our results in the complex geometry conditions.

Chapter 4 describes an alternative approach to deal with complex geometries, i.e. the Cartesian cut-cell Method. In particular, the treatment of the cells and points lying on and adjacent to the solid boundary and the boundary mesh distinguishing method are detailed. Error estimates of the cut-cell near the wall is investigated using finite difference approximations by polynomials and Richardson Extrapolation Methods. Due to the introduction of the Cartesian cut-cell, the more detail investigation is used in the boundary condition treatment. Chapter 5 gives the details of transformation algorithms (restriction and prolongation modification) and irregular boundary condition treatment.

Chapter 6 investigates the solution of the Navier-Stokes equations using the Cartesian cut-cell method for several standard cases, such as the laminar flow along a semi-infinite flat plate, a flow past a backward-facing step, and channel flow which is not aligned with coordinate axes. Exact solutions and other numerical results are used to validate the results. The extension of the Cartesian cut-cell method to more complex geometries is explained in Chapter 7. Curved channel flows (90^0 and 180^0 bend) (approximated by cut cell) have been used to investigate the performance of the complex boundary conditions in two-dimensional incompressible flow. Comparison of the solution of CC-PAMG, the numerical results of a commercial CFD code, CFX 4.2 [57],and the analytical solutions are made.

Finally, in Chapter 8, we close this thesis by presenting a summary of the work, a list of the conclusions that have been drawn, and a few suggestions as to how the

work could be usefully extended.

1.2 Overview of Grid Generation Methods

The task of obtaining solutions to the governing equations of fluid mechanics represents one of the most challenging problems in science and engineering. At the present time the mathematical formulations of the fundamental laws of fluid mechanics are governed by nonlinear Partial Differential Equations (PDE) which are of the three types: elliptic, hyperbolic, and parabolic. Classification of the three types is determined by the features of the physical problems. Typical examples in Fluid Dynamics are the Euler Equations and the Navier-Stokes Equations. Since analytical solutions of most nonlinear Partial Differential Equations are still not available, their solutions can only be formed by a numerical approach.

Great progress has been achieved during the last three decades in both Computational Fluid Dynamics (CFD) and applied mathematics, some of the achievements include the development of grid generation packages, adaptive multigrid algorithms, and difference schemes for solving Euler and Navier-Stokes Equations. However, there are still several barriers to overcome to obtain efficient and accurate solution for many large scale flow problems. Continued development of accurate, reliable and effective numerical methods and to extend these methods to complex industrial design and analysis applications is required.

Although the variety of complex flows that computational fluid dynamics researchers can analyse continues to increase, the solutions to much more complex flows are desired. Improved computer capacity has and will continue to have a large effect on the quality of solutions obtained. However, the recent improvements in the numerical solution algorithms are also important. These new numerical algorithms attempt to overcome two major obstacles to obtaining complex flow solutions; the geometric complexity of the solution domain for realistic problems and the existence of disparate length scales in the solutions.

Grid generation is a difficult task for complex geometric configurations. Many techniques still require user input to generate a grid for each new configuration. Modern techniques increasingly automate this process. Geometric complexity can be handled by more sophisticated grid generation schemes.

To overcome the disparate length scales, the grid should be allowed to adapt to the solution to ensure that high-gradient regions in the flow are not under-resolved and that low gradient regions in the flow are not over-resolved (Powell, [96]). Dominant local length scales can be smaller than the global flow length scales. Adapting the computational grid allows the grid spacing to respond to the local length scales of the flow. The adaptation scheme must detect the flow features and respond by increasing or decreasing the local resolution of the grid.

The numerical solution of fluid flow problems governed by the Navier-Stokes equations is generally accomplished using finite difference, finite volume, or finite element methods. The physical domain is mapped using a grid which should be more refined in regions of high gradients of the dependent variables. At present, the main grid generation approaches in computational fluid dynamics fall into either structured or unstructured grid categories.

For a complex boundary geometry, structured mesh methods are frequently based on the 'Chimera' philosophy of overlapping several meshes, each of which is specific to an appropriate section of the geometry. In such a case there is a high computational penalty which could be avoided by using a domain decomposition approach. Several techniques of this kind have developed, such as overlapping grid [24], zonal methods [100], and grid embedding techniques [84] .

However, this approach requires continuous identification of mesh intersection points from all overlapping meshes to transfer information between the various mesh patches. If any individual body geometry is very complex, the mesh generation re-

mains a difficult problem.

For unstructured meshes, the usual strategy is to use a global unstructured mesh and incorporate periodic local or global re-meshing to account for the moving boundaries or bodies. Although unstructured mesh methods can cope with complex geometries, they still need a significant amount of user intervention and encounter difficulties in preserving boundaries and reducing grid skewness. Furthermore, since the grid is unstructured, the change to the grid is purely local. The main difficulty is to determine when and where to interpolate while using adaptive refinement. Unstructured mesh methods may, in some cases, suffer from the effects of numerical diffusion.

In this thesis, we present a new, alternative approach for the prediction of steady incompressible flow-fields involving complex two-dimensional geometries. The method is based on Cartesian cut-cell techniques and does not at any stage involve any movement of the mesh. Rather, a stationary background Cartesian mesh is employed and the solid bodies are cut out of the mesh. In this way, the Cartesian cut-cell approach deals with complex flows around complicated geometries.

It has been our experience that the major obstacle to developing a Cartesian cut-cell method lies in formulating a general strategy that can cope with truly complex geometries. This method is quite simple, solid boundaries blank out areas of a background Cartesian mesh, and the resultant cut cells receive special attention during the integration of the flow solution. However, this simplicity of concept belies the obstacles that must be overcome in order to achieve a practical scheme. While these obstacles are far from insurmountable they are often perceived as stumbling blocks, hence the dearth of schemes which employ the Cartesian boundary method.

1.3 Structured Grid Methods

Discretisation of a domain can be accomplished either directly in the physical space or on the transformed computational space. The choice will primarily depend on the numerical scheme to be utilised, as well as the domains of solution. The finite difference equations approximating the partial differential equations are solved within a rectangular, equally spaced grid system. For non-rectangular physical domains, a coordinate transformation to computational space is required. The grid points are defined at the intersection of equally distanced parallel lines within the rectangular (2-D) or cubical (3-D) computational domain. There are corresponding grid points within the physical space established by algebraic relations or differential equations. The grid points can be easily identified and are usually designated by the indices, i, j, and k in an orderly manner along the grid lines. This type of grid is referred to as being structured, include two different structured grids: body-fitted and unfitted grids.

Before the choice of an unstructured grid is explained, it is first necessary to discuss the benefits and disadvantages of what is traditionally referred to as a structured grid approach. The later approach stores the data associated with the grid geometry and flow solution in logically ordered arrays. This has direct restrictions upon the topology of the grids that may be described and limits the connectivity of cells to their neighbours. For a structured grid approach, each cell has only one other cell lying on each of its faces. These local neighbour cells are always needed in a computation, for reconstructing the local solution gradients in a cell, and for a structured approach are found simply by incrementing or decrementing indices in the arrays. This simplification in the data structure has a large impact upon the simplicity of the resulting flow solver and due to this simplicity, can result in computationally efficient solution strategies. By locating all of the data in contiguous locations in memory, which is naturally done by storing the data in arrays, compilers can create very efficient codes, and the fast computation can result with using vector class supercomputers and cache usage.

Another benefit is the smoothness of the grid for simple domains. The grid smoothness is a direct consequence from the solution strategy used to form the grid. There are mainly two classes of methods used to generate body-fitted, and structured grid: algebraic based and partial differential equation based.

Algebraic based methods are more easily applied to complicated geometries, but typically requires a smoothing operation that is locally applied to the algebraically generated grid. The major advantage of this scheme is the speed with which a grid can be generated [41]. An algebraic equation is used to relate the grid points in the computational domain to those of the physical domain. This objective is met by using an interpolation scheme between the specified boundary grid points to generate the interior grid points. Clearly, many algebraic equations can be introduced for this purpose.

A grid generation scheme which has gained popularity is one in which PDEs are used to create the grid system [42]. In these methods, a system of PDEs is solved for the location of the grid points in the physical space, whereas the computational domain is a rectangular shape with uniform grid spacing. These methods may be categorised as an elliptic, parabolic or hyperbolic system of PDEs. Parabolic and hyperbolic grid generators have some very interesting properties. The benefits of this approach come from the smoothness guaranteed by the elliptic equations. Since solutions to elliptic equations satisfy certain differentiability properties and guarantee satisfaction of a local maximum principle, smooth grids can be obtained and grid line crossing can be eliminated.

Comparison of body-fitted grid and unfitted Grid, the transformed grid is boundary-fitted and smooth at boundary layers. However, due to the transformation from physical space to computational domain, the governing equations are more complex and need more memory storage (grid matrices). Contrary to the transformed grid, the unfitted grid has a simple governing equation which is easy to solve and less computational memory, but it is difficult to treat the complex boundaries. Thanks to the Cartesian cut-cell algorithm which can overcome this problem.

The biggest difficulty with structured grid methods is grid generation. Grids still cannot be generated for arbitrary complex geometries in a totally automatic way. The reason is that the grid data structures are simple. By limiting cells to have a set number of sides, the domain with complicated geometries must be decomposed into logical zones or patches. Each patch needs to have the same number of sides as the sub-domain cells which make up the domain. In two dimensions, this means four-sided patches and in three dimensions blocks of six sides. Then, in each zone, a grid generation approach is applied and the resulting iso-coordinate lines joined, yielding a grid network and the computational volumes. Depending on the flow solver to be used, the grid lines may need to be connected across the patches, increasing the difficulty of the problem. This sub-domain approach becomes increasingly complicated as the complexity of the geometry increases. This downside of structured grids has led researchers in recent years to favour unstructured grids in order to compute flows about complicated geometries.

1.4 Unstructured Grid Methods

In addition to finite difference schemes, two other numerical schemes are available for the solution of conservation laws. These schemes are finite volume schemes or finite element schemes. Both of them are integral methods, that is to say, the original differential equations are integrated on the physical domain and subsequently, are solved numerically. Therefore, the grid system for the finite volume or finite element schemes are usually generated directly within the physical space. There exist various choices in the selection of the volumes or elements. Thus, the domain of solution is usually divided into triangles or quadrilaterals in two-dimensions, whereas pyramids or tetrahedrals are used in three-dimensions. It is obvious that the grid points cannot, in general, be associated with grid lines. Therefore, the identification of the grid points must be individually specified. This grid system is known as an unstructured grid system.

Unstructured grid methods can provide grids for much more complex geometric configurations, but may not be able to handle arbitrarily complex geometries [73]. Traditional unstructured grids that are in use by the aerodynamic and other communities are typically based upon triangular or tetrahedral elements. This means that the grid generation process specifying the bounding surfaces and filling the domain with these elements. The volume grid is generated so that the cells on the boundaries of the domain are coincident with the boundaries. In this sense, the volume grid is constrained by the boundary discretisation.

The benefits of the unstructured grid methods is that there is no mapping from the physical space to a simple computational space. The main advantage of the unstructured grid is that it can be used easily to fit irregular, simply-connected domains, as well as multiply-connected domains. The unstructured grid also can be coupled with grid refinement techniques for the adaptive methods. However, unstructured grid generation is more difficult to program, that is the programmer needs a sound background in the data structure arrangement and experience in the data book-keeping skills. Also this implies a computational cost.

There are three kinds of methods that are currently being used to generate the volume grid for unstructured meshes [75]. The advancing front methods is based upon starting at the boundaries and advancing a 'front' of vertices into the computational domain. The new vertices are connected with existing nodes (triangulated) and the front is advanced until the entire domain is covered. Grid size and smoothness can be controlled by source terms which vary in space according to a user specified criteria.

Another sophisticated grid generation scheme is the Delaunay triangulation [3] which is based upon triangulating a set of points and this is also used quite frequently. The basic idea is to consider a point for triangulation, and connect it to its neighbours in a way that satisfies certain geometric constraints. If the existing local points do not satisfy the geometric criteria, a point or set of points are locally added

to satisfy the criteria. The Delaunay criterion is shown to have many important implications regarding locality of neighbours, interpolation properties, smoothness of the grid and is shown to satisfy certain geometric properties that are inherent.

The third grid generation method is the 'front-Delaunay' method with combination of advancing front and Delaunay method [75]. It has the benefits of the two above methods and overcomes some disadvantages. There is still much research on the unstructured grid generations schemes.

Unstructured grids are ideally suited for grid embedding. In areas where adaptation is needed, a cell is divided or replaced, by a number of smaller cells. Since the grid is unstructured, the change to the grid is purely local. The main difficulty is to determine when and where to refine. A number of different refinement criteria may be used which in some way use an estimate of the solution error. Then a threshold for refinement is set either by experience or by information obtained from the distribution of the error.

1.5 Cartesian Grid Methods

The Cartesian cell-based approach presented here performs an important task and generates a volume grid automatically when given the functional or discrete description of multiple bodies and domains. In addition, due to the simple data structure used to store the grid and flow data, adaptive mesh refinement is a natural extension of the approach. This handles the possibility of achieving grid converged solutions upon automatically generated grids.

Advantages of using the unfitted Cartesian meshes are simple flux formulations, simplifications in the data structure, and relatively easy grid generation. In addition, Cartesian cells lead to fortuitous cancellation of truncation errors not occurring on less regular meshes. It also has an efficient internal formulation which is better than the

body-fitted structured or unstructured grid methods. The disadvantage that arises in using Cartesian cells is due to the treatment of complicated boundary conditions. However, there are some techniques such as the adaptive multigrid methods that can overcome this problem. Cartesian grids have a couple of difficulties to overcome: the difficulty of resolving high curvature regions of a body and the introduction of cut cells that are a small fraction of the size of un-cut cells.

1.6 Cartesian Cut-cell Methods

The use of Cartesian based cells to discretize a domain is not a new idea. Indeed, it is the simplest possible discretisation for domains which are square or rectangular. The novelty of the unfitted grid approach arises from the application of Cartesian-cells to non-square domains. For non-simple geometries, all Cartesian based approaches must perform some type of a specialised procedure on the boundaries to account for the non-alignment of the boundary with cell geometry. The particular strategies of these special boundary procedures are dependent upon the type of algorithm used on the interior of the domain, and are naturally dependent upon the equations that are being solved.

The small cut-cell issue is a significant difficulty arising from the use of the Cartesian grid method to arbitrary boundary conditions. The small cut cells not only lead to inaccuracies in the flow by becoming decoupled from the rest of the flow, but to severe restrictions on the cell time-step [34]. This penalty occurs at boundary geometries where the cells are been cut. At these regions, the grid adaptation is very important to reduce the inaccuracies. As the capacity of supercomputers increases and new numerical solution algorithms are developed, this disadvantages will be overcome.

Compared to other types of grid, the Cartesian cut-cell method appears straightforward. Solid bodies merely blank out areas of a background, Cartesian mesh. This

gives rise to three classes of mesh cells. Namely, cut cells which lie along the surface of a body, solid cells which lie wholly outside the flow field and un-cut cells which lie within the flow field. But a cut-cell may be further categorised as one of several types, depending on the numbers and relative positions of the intersection of the flow field with the cell.

So, given an arbitrary set of bodies, the process of determining the exact nature of a cell so as to be able to perform a finite volume flux integral, is not trivial. Moreover, since we employ an adaptive grid this cell-type information cannot be calculated as a pre-processing of each calculation; it must be calculated each time the grid is adapted. Therefore, considering that a typical calculation involves parts of grid adaptions, it is imperative that the calculated process be efficient.

1.7 Objectives

Incompressible flow modelling is an important area of general industrial interest which has received significant attention from researchers for many years. In this area, computational techniques have been developed which allow for the modelling of complex geometrical configurations by generally employing unstructured grids. A comparatively small amount of effort has been devoted to the development of methods which based upon a velocity-pressure formulation and equal order interpolation.

After discussing the advantages and disadvantages of each individual grid approach, perhaps the winning candidate for the future computational fluid dynamics will be a hybrid structured-unstructured mesh, where the goal is to maximize the advantages and minimize the disadvantages of each approach. Key advances in hybrid grids are now being made by the Cartesian cut-cell method combined with adaptive multigrid methods, which uses an overall structured 'parent' grid that can be locally refined by lower level 'child' grids that can still be further refined (if required) by still lower level grids to as fine a level as desired for resolving both internal features of

the flow and boundary geometries using Cartesian cut-cell to approach the boundary conditions.

The primary objective of this thesis is to develop an efficient and accurate method of predicting steady incompressible laminar flow over arbitrary, complex two dimensional geometries which are encountered in the engineering industry and scientific research. To facilitate this, it has been necessary to develop a grid generation technique (Cartesian cut-cell algorithm) capable of dealing with arbitrary configurations and also an appropriate flow solver. A novel multigrid algorithm has been developed by Thompson et al. [118]. The relaxation scheme avoids the introduction of Brunt-Vasaila oscillations which limit the performance of classical, segregated approaches. This method appears to be reasonably efficient and robust, converging over a range of physical parameters from zero initial guess.

In the present study, steady state solutions for the two dimensional semi-infinite flat plate flow, the backward facing step flow, the non-coordinate axis aligned channel flow as well as curved channels are obtained. The method is not purely restricted to these geometries and may, in principle, be applied to any two-dimensional configuration.

Chapter 2

A review of relevant literature

2.1 Introduction

Needless to say, elliptic, parabolic, and hyperbolic partial differential equations are at the heart of most mathematical models used in engineering and physics, giving rise to extensive computations. Often the problems that one would like to solve exceed the capacity of even the most powerful computers, or the time required is too great to allow inclusion of advanced mathematical models in the design process of technical apparatus, from microchips to aircraft, making design optimisation more difficult. Multigrid methods are a prime source of important advances in algorithmic efficiency, finding a rapidly increasing number of a users. Unlike other known methods, multigrid offers the possibility of solving problems with N unknowns with O(N) work and storage, not just for special cases, but for large classes of problems.

Multigrid methods offer an alternative to implicit methods for efficiently solving large problems, while incurring low additional memory overheads. A notable property of a well formulated multigrid algorithm is that the number of multigrid cycles required to achieve a given level of convergence is independent of the resolution of the mesh. Multigrid methods are also very powerful, in that they may be applied to linear and non-linear problems.

The basic idea behind all multigrid strategies is to accelerate the solution of a set of fine grid equations by computing corrections on a coarser grid. The motivation for this approach comes from an examination of the error of the numerical solution in the frequency domain. The rate of convergence of basic iterative method can be improved with multigrid methods. The basic observation is that long wavelength modes decay slowly, short wavelength modes are reduced rapidly. The essential multigrid principle is to approximate the smooth (long wavelength) part of the error on coarser grids. The non-smooth or rough part is reduced with a small number of iterations with a basic iterative method on the fine grid. In fact, the convergence rate of single grid iterative methods usually consists of a rather rapid initial residual reduction phase, which gradually develops into a much slower residual reduction phase, corresponding to a situation where all high-frequency errors have been eliminated and low-frequency errors dominate, as shown in Figure 2.1.

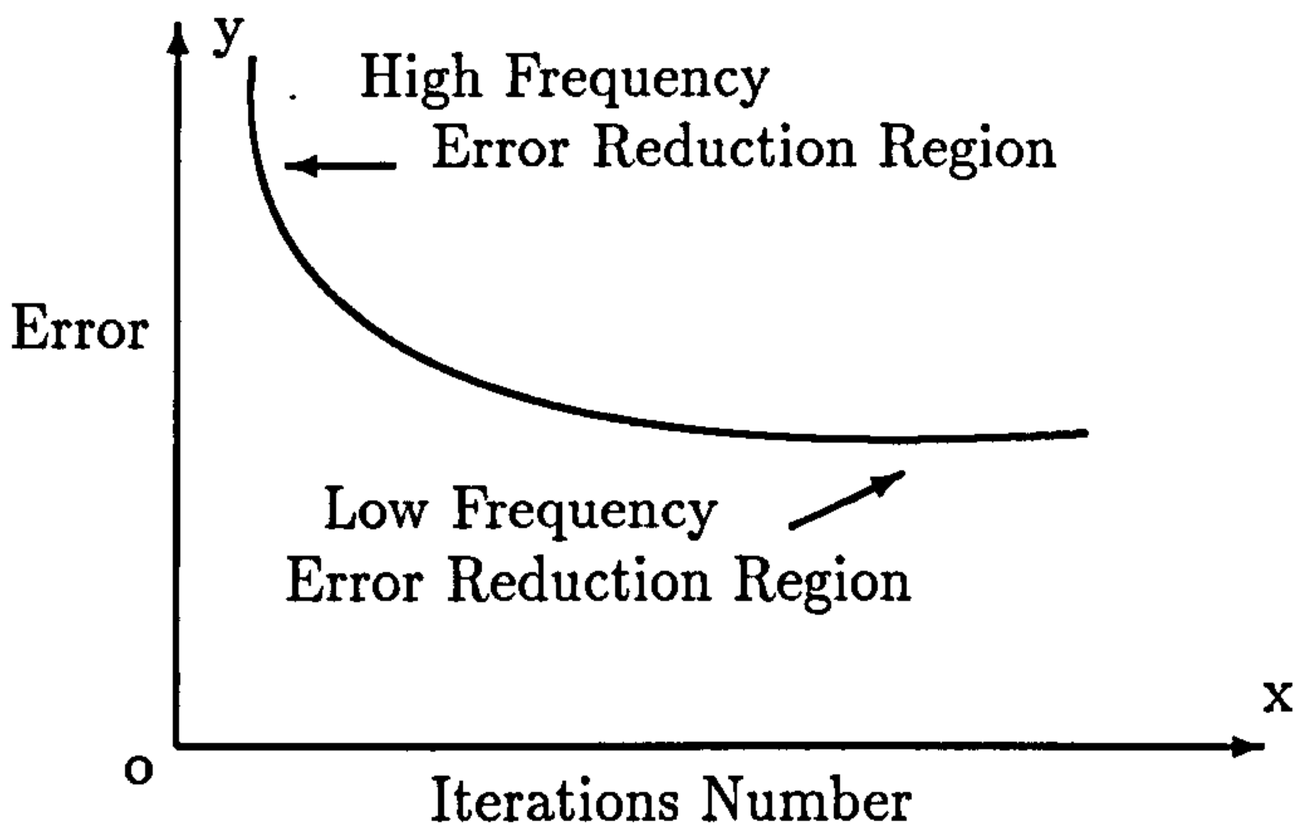


Figure 2.1: Typical convergence characteristics of iterations schemes (Mavripilis, [74]).

Multigrid strategies are generally considered as convergence acceleration techniques, rather than solution methods themselves. In fact, they may be applied to any existing relaxation technique. The success of the overall solution strategy depends on

the close matching between the bandwidth of errors which can be efficiently smoothed on a given grid using the particular chosen relaxation strategy, with a careful construction of a sequence of coarse grids, in order to represent the entire error frequency range.

2.2 Historical Development of Multigrid Methods

Since Brandt [14] published the first practical solutions of Multigrid Methods, more and more papers were published in this area and later extended to other related research areas. As we know, multigrid methods have been developed for nearly three to four decades. In what probably was the first 'true' multigrid publication, Fedorenko [44] developed an original idea to solve the algebraic equations. The idea is to solve algebraic equations approximately using a coarser grid, where the problem on the coarser mesh is solved by using still coarser grids in a similar fashion. Thus each computational cycle for the finer level contains several cycles for the coarse level, and so forth. He also formulated a multigrid algorithm for the standard five-point finite difference discretisation of the Poisson equation on a rectangular domain, that such an iterative method has a rate-of-convergence $0 \leq q < 1$, where q is independent of the mesh-size, proving that the work required to reach a given precision is O(N). It seems to be the only known method with such a property [130].

This basic contribution was extended by Bachvalov [1], using a change of variable in a rectangle for the solution of a network approximation of Poisson's equation. The combination of this method with Fedorenko's original idea, is put forward for the solution of a network approximation of Poisson's equation in a square. It was generalised to the central difference discretisation of the general linear elliptic partial differential equation in $\Omega = (0,1) \times (0,1)$ with variable smooth coefficients. The basic multigrid methods and the proof of the convergence are based on general considerations which suggest the possibility of using the method in the case of arbitrary, and in particular non-linear, equations and other classes of boundary conditions. The theoretical work

estimates were pessimistic, and the method was not put into practice in the middle of sixties.

The first practical results were reported in a pioneering paper by Brandt [14], who published another paper in [15], clearly outlining the main principles and the practical utility of multigrid methods, which drew wide attention and marked the beginning of a rapid development. He first called this new algorithm the Multi-Level Adaptive Technique (MLAT). The main ingredients of MLAT are (a) adaptive discretisation and (b) multi-level iterative procedure in which coarser grids constantly participate in solving the equations on finer grids. It is claimed that this method yields an efficient discretisation, better convergence rates, stability, self-correcting, naturally suitable for non-linear problems, demands minimal storage, does not require high computer precision, and is very suitable for parallel processing.

The multigrid method was developed independently by Hackbusch in 1976, who laid firm mathematical foundations and provided reliable methods (Hackbusch [50, 51, 49]). A report by Frederickson [46] describing an efficient multigrid algorithm for the Poisson equation led Wesseling to the development of a similar method for the vorticity-stream function formulation of the Navier-Stokes equations, resulting in an efficient method [128, 129]. He also published several papers on the rate of convergence of multigrid methods.

At first there was much debate and scepticism about the true merits of multigrid methods, which only changed after sufficient satisfactory results were obtained. This led a number of researchers to the development of stronger and more transparent convergence proofs (see Hackbusch [54] for a survey of theoretical developments). Other authors have tried to spread confidence in multigrid methods by providing sufficient and reliable computer programs, as much as possible of 'black-box' type, for discretisations of PDEs. The 'multigrid guide' of Brandt [16, 17] was provided to give guidelines for researchers writing their own multigrid programs.

For an introduction to the application of multigrid methods to these subjects, see Hackbusch [53, 54] and Brandt [19]. For important recent advances in the field of integral equations, see Brandt and Lubrecht [21]. A recent publication on parabolic multigrid method can be found in Murata et al. [89]. Although most theoretical work has been done in a variational framework, most applications used finite volumes or finite differences techniques.

Vanka [121] developed a new method solve the Navier-Stokes equations in primitive variables using a coupled block-implicit multigrid procedure. The procedure is applicable to finite-difference formulations using staggered locations of the flow variables. A smoothing technique called symmetrical coupled Gauss-Seidel is proposed and is observed to provide good smoothing rates. In terms of residuals and corrections at node (i,j), the equations can be arranged in a block structure as follows:

$$\begin{bmatrix} (A_c^u)_{i-1/2,j} & 0 & 0 & 0 & 1/\rho\delta x \\ 0 & (A_c^u)_{i+1/2,j} & 0 & 0 & -1/\rho\delta x \\ 0 & 0 & A_c^u)_{i,j-1/2} & 0 & 1/\rho\delta y \\ 0 & 0 & 0 & A_c^u)_{i,j+1/2} & -1/\rho\delta y \\ -1/\delta x & 1/\delta x & -1/\delta y & -1/\delta y & 0 \end{bmatrix} \begin{bmatrix} u'_{i-1/2,j} \\ u'_{i+1/2,j} \\ v'_{i,j-1/2} \\ v'_{i,j+1/2} \\ p'_{i,j} \end{bmatrix} = \begin{bmatrix} R^u_{i-1/2,j} \\ R^u_{i+1/2,j} \\ R^v_{i,j-1/2} \\ R^v_{i,j+1/2} \\ R^c_{i,j} \end{bmatrix} \tag{2.1}$$

Multigrid principles (Appendix A) are much more widely applicable than just to the numerical solution of differential and integral equations. Applications in such diverse areas as control theory, optimisation, pattern recognition, computational tomography and particle physics are beginning to appear. For a survey of the wide ranging applicability of multigrid principles, see Brandt [18, 19].

2.3 Development of Cartesian Cell Methods

In recent years there has been an upsurge of interest in the development of numerical methods based upon the use of unstructured grids and unfitted Cartesian grids for the solution of the equations of compressible flow. This interest has been largely driven

by the requirements of the aerospace and other fluid engineering industries for the provision of a procedure which would allow the analyst to rapidly analyse high and low speed flows over configurations of complex shape. As a result of these efforts, unstructured grid methods have now reached a relatively high level of sophistication, both in terms of grid generation procedures and flow solver accuracy.

There are a number of applications to which Cartesian based algorithms have been applied, and depending upon the simplicity or complexity of the governing solution procedure, different approaches have resulted. The Cartesian based approach has been used successful for the solution of the full potential equation by a number of researchers. One of the first applications was made by Purvis and Burkhalter [97], where they used a finite-volume procedure to solve the full potential equation, and took advantage of the flux formulation at cut-cell boundaries to simplify the boundary procedures. The background mesh was regular, and stored in a single array. Since the flux through the body face is zero by construction, the only contributions to the cell update come through the exposed faces of the cut cells. A more efficient relaxation strategy was used by Wedan and South [127], who used a line Gauss-Seidel relaxation scheme instead of the point Gauss-Seidel approach. Neither of these approaches used local refinement.

For transonic flows, weakly-shocked flows and un-shocked flows, solution of the full potential equation can result in excellent predictions of the flow field. But, when the shock strength are increased, or the flow is rotational, it is necessary to solve the complete Euler equations. Unsteady, adaptively refined solutions of the Euler equations have been obtained by Berger and Oliger [13], Berger and Colella [9], and Quirk [98] for Cartesian geometries. These calculations demonstrate the excellent results that can be obtained for highly complicated shock hydrodynamic problems in performing the mesh refinement and flux computations. The flexibility and utility offered for non-Cartesian bodies that was shown for the full potential equation was first extended to the Euler equations by Clarke, Salas, and Hassan [27], where non-refined, finite-volume solutions about multi-element airfoils were calculated using

this approach. For steady inviscid flows, adaptively refined solutions using an upwind finite-volume approach are shown by De Zeeuw et al. ([38, 35]) and by Coirier and Powell [28]. The algorithm creates an initial uniform mesh and cuts the body out of that mesh. The mesh is then refined based on body curvature. Next, the solution is converged to a steady state using a linear reconstruction and Roe's approximate Riemann solver. Solution adaptive refinement of the mesh is then applied to resolve high-gradient regions of the flow. The numerical results presented show the flexibility of this approach and the accuracy attainable by solution based refinement in transonic flow.

The extension of the Cartesian-cell based approach to the Navier-Stokes equations may be found in several papers. Thompson et. al ([118, 113, 117]) used the Cartesian multigrid approach algorithms to solve a buoyancy-induced flow problem, and extended it to adaptive solutions in the Navier-Stokes equations. However, this paper did not consider non-Cartesian boundary conditions. A multigrid algorithm based upon a novel relaxation scheme has been developed which handles energy-momentum coupling correctly. Numerical experiments have been performed to show that this approach is reasonably efficient and robust for a range of Rayleigh numbers and a variety of cycling strategies. The basic algorithm is a multigrid method based on a robust, box-based smoothing step. Its most important feature is the incorporation of automatic, dynamic mesh refinement. This refinement strategy is completely local, and has the important consequence that boundary layers and other regions of sharp transition do not 'steal' mesh point from surrounding regions of smooth flow, in contrast to moving mesh strategies where such 'stealing' is inevitable. Thompson and Lezeau [63] has extended this algorithm to adaptive quasi-Newton coupled multigrid solver for the simulation of incompressible unsteady multiphase flows. The methodology has proven highly successful for multiphase flows, leading to solution algorithms which are robust, efficient, and accurate in comparison to other solvers.

The extension of the Cartesian cell-based approach to the compressible Navier-Stokes equation was developed in the Ph.D. thesis by Quirk [98] with a finite volume

approach. The work in the finite volume area by Quirk had its major emphasis upon unsteady, upwind numerics, and concentrated on obtaining high quality unsteady shock hydrodynamic solutions. The extension to the Navier-Stokes equations was brief, and was demonstrated for only a few simple test cases. The work performed carefully extended the finite-volume formulation of the Cartesian cell based approach to solving the Navier-Stokes equations, and gives a more complete analysis of the approach and demonstrations of its capabilities.

The primary grid system is an Cartesian grid, automatically generated in three dimensions by Karman [60] using recursive cell subdivision. This grid system is sufficient for computing Euler solutions about extremely complex 3D geometries. A secondary grid system, using triangular-prismatic elements, may be added for resolving the boundary layer region near surfaces of solid bodies for Navier-Stokes analysis. A description of the methods used to generate the grids and solve the governing equations is provided, demonstrating the ability of the method to produce accurate solutions about complex geometries with very little pre-processing required by the user.

2.4 Development of Cartesian Cut-cell Methods

The extension of the unfitted grid approach to the Euler equations may be found in several papers. For adaptively refined flows about non-Cartesian geometries using the Cartesian based approach, there are the works by Berger and Le Veque ([12], [10], and Epstein et al. [43]). Quirk [99] and Chiang et al. [26] have demonstrated the excellent flow feature capturing capability of the Cartesian approach using upwind-based, finite-volume strategies in their work for unsteady, adaptively refined flows past non-Cartesian geometries. In this paper, Quirk developed the so-called Cartesian Boundary Method for dealing with arbitrarily complex two-dimensional geometries. The Cartesian Boundary Method is quite simple. Solid bodies blank out areas of a background Cartesian mesh, and the resultant cut cells are singled out for spe-

cial attention. Results are presented which demonstrate that this new algorithm can match not only the accuracy of results produced using body-fitted grids, but also the geometric flexibility exhibited by unstructured grid schemes.

A conservative interface algorithm for overlapping (Chimera) grids is extended to overlapped structured/unstructured grids by Wang and Yang [125] and Wang [124]. This extension is straightforward and efficient due to the fact that flow variables are not required to interpolate between grids. With the introduction of unstructured grids into Chimera, geometrically very complex components can be modelled easily. However, with the introduction of unstructured grids, the computational efficiency is compromised by the use of indirect addressing

Pember et al. [95] introduced the Godunov method[1] to deal with the adaptive Cartesian grid method with the Navier-Stokes equations and obtain good results. Coirier and Powell [29] developed and tested a Cartesian cell-based approach for adaptively-refined solutions of the Euler and Navier-Stokes equations in two dimensions. Grids about geometrically complicated bodies are generated automatically by recursive subdivision of a single Cartesian cell encompassing the entire flow domain.

The use of the Cartesian-based approach in three dimensions was investigated by Melton et. al ([80] and [79]), using ideas for the cut-cell generation borrowed from computer aided design and computer graphics, shows promise by providing a common way of describing the geometry of the problem. Improvements to flow field grid generation algorithms; geometry representations, and geometry refinement criteria were demonstrated, including details of a procedure for correctly identifying and resolving extremely thin surface features. An initial implementation of automatic flow field refinement were also presented.

Yang et al. [133] presented the numerical simulation of unsteady, axisymmetric

[1]Godunov [48] suggested that *exact solutions* of the Euler equations for a *local* region of the flow be *pieced together* to synthesise the *general* flow field.

compressible flow involving both fixed and separating bodies. A Cartesian cut-cell mesh approach and a high resolution, upwind finite volume scheme are used. The method has been applied to muzzle blast and related flows with rapidly moving shocks and separating bodies and has been validated by recourse to relevant experimental data. The computed results suggest that this approach can provide an accurate and effective numerical prediction tool for the simulation of unsteady compressible flow. The method presented here is valid for unigrid conditions and is not extended to multigrid methods.

A year later, Yang et al. ([134], [135]) based on the Cartesian cut-cell approach and multi-dimensional high resolution upwind finite volume scheme, could cope with static or moving body problems having arbitrarily complex geometries. They extended the Cartesian cut cell approach and finite volume scheme from static body problems to the moving body problems. A cell merging technique has been developed to maintain numerical stability in the presence of arbitrarily small cut cells and to retain strict conservation at moving boundaries. They introduced the Cartesian cut-cell approach and finite volume method in the compressible flow, However, extensions to adaptive multigrid methods is not touched. There still left some exciting interesting research to be made.

The current thesis introduced the Cartesian cut-cell method in incompressible flow and developed a novel algorithm to cope with the complex boundary conditions. Yang et al. [133] first introduced the term 'Cartesian Cut Cell' in their paper in compressible flow with single grid, the algorithm is different with what is presented here. For the adaptive multigrid method, there are several papers ([12] , [43], [99], [26], [95], [29]) which are published in the field of compressible flow. Incompressible flows have different properties (i.g. the special mass conservation requirement) which we addressed. The application of Cartesian cut-cell methods to incompressible flows is a novel aspect of the current research.

2.5 Commercial CFD Package: CFX 4.2

Computational fluid dynamic packages are becoming an integral process in the design and simulation of fluid mechanics. It is claimed that computer simulation of fluid flow helps reduce development time-scales and cost by replacing laborious and lengthy experiments with numerical simulation of the process, and by providing an accurate insight into fluid flow phenomena even when experiments cannot be conducted. Potential disturbances due to the measuring device would be eliminated and the potentiality of easily interrogating all physical variables would provide further beneficial information.

The first requirement is that our adaptive cut-cell results give accurate answers. The necessity of easily obtaining detailed numerical information means that commercial CFD packages provide an ideal solution to validate our algorithm. We used a commercially available package produced by AEA Technology Plc. (The software is entitled: CFX 4.2). This software package is used in simulating fluid flows for two-dimensional and three-dimensional cases.

CFX is a powerful fluid engineering software tool which can be use to simulate many types of industrial flow and heat transfer processes. The CFX 4.2 flow modelling software package is a suite of programs for the prediction of laminar and turbulent flow, heat transfer, and chemical reactions. Version 4.2 extends the geometric capabilities of the code by using the facility of multi-block, or block-structured grids. Through this period, other enhancements have been incorporated into the software.

Chapter 3

Governing Equations and the Discretisation

3.1 Introduction

Dynamical relations describing the motion of a fluid are concerned essentially with the response of a specified piece or mass of the fluid to external influence. It is useful therefore to develop ways of describing the physical history of a material portion of fluid which may be undergoing distortion as well as change of position. Physical models for fluid flows are usually formulated, in an Eulerian framework, in terms of partial differential equations. They are the mathematical statements of three fundamental physical principles (conservation laws) upon which all of fluid dynamics is based. they are: the mass of a fluid is conserved, the rate of change of momentum equals the resultant of the force on a fluid particle, and the rate of change of energy is equal to the sum of the rate of heat addition to and the rate of work done on a fluid particle. By contrast, numerical analysis codes solve discrete algebraic equations. Discretisation is the process by which a correspondence between the two representations is established and is therefore the very first stage in the numerical simulation of a physical phenomena. Finite difference analysis and finite element analysis, for instance, are means of replacing the continuous problem by a discrete one whose solution is close to the solution of the original problem.

This chapter discusses the governing equations, in discrete and continuous form, which are solved by the PAMG code as well as the specific methodology used to perform the discretisation. Discrete difference equations are based on finite difference or finite volume method to the governing partial differential equations. This ensures, amongst other things, that the principle of conservation is automatically satisfied for the discrete problem as well as the continuous problem. The discretisation is performed on staggered grids which are well suited to the computation of incompressible flows. The hybrid scheme introduced by Spalding [109] is used to derive discrete equations which can be accurate up to second order without generating non-physical oscillations. The use of staggered finite volume grids and hybrid schemes are design choices which are key factors for the robustness and efficiency of the solution algorithms.

3.2 Hybrid Schemes

Hybrid schemes are based on an observation made by Spalding [109] that the best discretisation of a differential equation involving convective and diffusive terms depends on the relative magnitude of these terms. He considered the ordinary differential equation:

$$c_p G \frac{dT}{dx} - \lambda \frac{d^2T}{dx^2} = 0, \tag{3.1}$$

which describes the steady state transfer of heat in a one-dimensional moving medium. By introducing $\xi = x/L$ where L is the length of the domain, and the Peclet number

$$P = \frac{c_p G L}{\lambda},$$

the equation can be rewritten in the form:

$$P \frac{dT}{d\xi} - \frac{d^2T}{d\xi^2} = 0. \tag{3.2}$$

He shows that applying central differencing to both the convective and the conductive term is only acceptable if the magnitude of the Peclet number is small (*i.e.* $\mid P \mid < 2$). If $\mid P \mid$ is larger, i.e. when the convective effect is predominant, it is better to discretize the convective term by an upwind scheme. Upwind differencing is adapted to situations when convection is the dominant factor because it implies that the value of a convected quantity at some point is equal to its value immediately upstream. In fact, Spalding shows that if the convective part is dominant, it is better to ignore the diffusive flux altogether. Hybrid schemes are discretisations which mix central and upwind differencing in a way which is dependent on the relative strength of the convective and diffusive terms.

In steady incompressible flows, the Peclet number is replaced by the Reynolds number (Re); in buoyancy driven flows, the Prandtl number is also relevant. All these non-dimensional quantities control the relative importance of convective and diffusive effects. They are based on a characteristic length of the computational domain.

For the design of a hybrid scheme, the characteristic lengths are taken to be the discretisation steps. In this way, cell numbers are introduced. For instance the Reynolds number could be defined as

$$Re = \frac{\rho u L}{\mu}, \tag{3.3}$$

and the cell-Reynolds number is:

$$Re_c = Re\Delta\xi$$

.

The numerator in the above equation measures the magnitude of the convective transport while the denominator is proportional to the viscous forces. The analysis for fluid flow is complicated by the fact that the convective terms become non-linear.

If the magnitude of the controlling cell number is smaller than 2, both the convective and diffusive flux are discretized by second order central differencing. If the magnitude exceeds 2, the diffusive fluxes are ignored, and the convective fluxes are discretized using first order upwind differences. With hybrid differencing, we have to some extent, the concept of a local discretisation.

Usually, finite differences formulas are obtained by expansions in Taylor series. In contrast, hybrid schemes, although they are of finite difference type are obtained by a finite volume analysis. The main advantage is that numerical conservation is automatically satisfied over a structured domain centred around the variable for which a formula is sought. Typically, the domain chosen has the same size as a grid cell.

Spalding [109] gives an example of the process in one dimension. The scheme he obtains is defined in terms of the solution. However, simpler schemes can easily be derived. The heat flux at any point is the crucial quantity for deriving finite difference formula in this way. It is obtained by integrating the left hand side of the Equation (3.2) giving

$$Q(\xi) = \int \left[P\frac{dT}{d\xi} - \frac{d^2T}{d\xi^2} \right] d\xi = PT - \frac{dT}{d\xi} = constant. \tag{3.4}$$

A finite difference formula can be obtained from approximation of the fluxes using the principle of the conservation of heat, i.e. the fact that through a control volume, the total flux of heat is equal to the heat sources in the control volume.

A numerical approximation of the flux at middle distance between two points, ξ_1 and ξ_2 on the discrete grid can be obtained as a function of the temperature at the two points. If $Q_{1+1/2}$ and $Q_{1-1/2}$ are the flux at the points $(\xi_1+\xi_2)/2$ and $(\xi_2+\xi_3)/2$, a finite difference formula is obtained for T_2 as a function of T_1 and T_3 by stating

$$Q_{1-1/2} = Q_{1+1/2},$$

the first derivative of (3.2) is approximated by central differences:

$$\frac{dT}{d\xi}\left(\frac{\xi_1+\xi_2}{2}\right) \approx \frac{T_2-T_1}{\delta\xi}.$$

There are two possibilities for the approximation of the convective term PT: centred differencing or upwind differencing

If central differencing is chosen,

$$T_{1+1/2} \approx \frac{T_2+T_1}{2},$$

The heat fluxes around ξ_2 are therefore:

$$Q_{1-1/2} = P\frac{T_1+T_2}{2} + \frac{T_2-T_1}{\delta\xi},$$

$$Q_{1+1/2} = P\frac{T_2+T_3}{2} + \frac{T_3-T_2}{\delta\xi}.$$

Equating the two fluxes, and dividing by $\delta\xi$ yields:

$$P\frac{T_3-T_1}{2\delta\xi} - \frac{T_3-2T_2+T_1}{\delta\xi^2} = 0. \tag{3.5}$$

Working from Taylor's series gives an equivalent result.

If upwind differencing is chosen, the direction in which the heat flows is important: if the heat flow from ξ_1 to ξ_3, that is if $T_1 > T_2 > T_3$, we take

$$T_{1-1/2} \approx T_1, \qquad T_{1+1/2} \approx T_2,$$

The fluxes are:

$$Q_{1-1/2} = PT_1 + \frac{T_2-T_1}{\delta\xi},$$

$$Q_{1+1/2} = PT_2 + \frac{T_3-T_2}{\delta\xi},$$

which results in the following discretisation:

$$P\frac{T_2-T_1}{\delta\xi} + \frac{T_3-2T_2+T_1}{\delta\xi^2} = 0. \tag{3.6}$$

If the heat flows in the other direction,

$$T_{1-1/2} \approx T_2, \qquad T_{1+1/2} \approx T_3,$$

The fluxes are:

$$Q_{1-1/2} = PT_2 + \frac{T_2 - T_1}{\delta\xi},$$

$$Q_{1+1/2} = PT_3 + \frac{T_3 - T_2}{\delta\xi},$$

which results in the following discretisation:

$$P\frac{T_3 - T_2}{\delta\xi} - \frac{T_3 - 2T_2 + T_1}{\delta\xi^2} = 0. \tag{3.7}$$

We also obtain standard schemes. The fundamental concept is that the direction of the heat flow at ξ_2 is given by the sign of P: physically, P is proportional to the mass flux rate of material in the positive x-direction. If $P > 0$ the heat flow from ξ_1 to ξ_3. If $P < 0$, the heat flow from ξ_3 to ξ_1.

Hybrid differencing is obtained by choosing different expressions for the fluxes according to the value of P:

(1) if $|P| < 2$, convective effects are not dominant and centred differences are used. In other words, we take

$$Q_{1+\frac{1}{2}} = P\frac{T_1 + T_2}{2} + \frac{T_2 - T_1}{\delta\xi}$$

(2) if $P > 2$, the diffusive term is ignored because convective effect are dominant. Because heat flows from ξ_1 to ξ_2, we take

$$Q_{1+\frac{1}{2}} = PT_1$$

(3) if $P < 2$, the diffusive term is also ignored but as heat flows from ξ_2 to ξ_1, we take:

$$Q_{1+\frac{1}{2}} = PT_2$$

3.3 Discretisation of the steady Navier-Stokes Equations

3.3.1 The Steady State Navier-Stokes Equations

In the context of incompressible viscous flows, the momentum is transferred by the mechanisms of convection and diffusion. The governing equations are therefore qualitatively equivalent to the energy equation. In two dimensions, they can be expressed in terms of the stream function: for momentum as well as energy, the convecting agent is the fluid velocity. The thermal diffusivity Γ is replaced by the kinematic viscosity ν.

The two-dimensional steady state flow of an incompressible viscous fluid is governed by the following form of the Navier-Stokes equations:

- **Conservation of mass:**

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{3.8}$$

- **Conservation of horizontal momentum:**

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = -\frac{\partial P}{\partial x}. \tag{3.9}$$

- **Conservation of vertical momentum:**

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) = -\frac{\partial P}{\partial y}. \tag{3.10}$$

These equations are discretized using a hybrid finite difference scheme, which employs central differences on the convection and diffusive terms. A standard staggered mesh is overlaid on the computational domain, where the velocities are associated with the edges of the cells, and the pressures are associated at the centres of the cells. A finite volume approach is followed with the hybrid scheme, the mesh is uniform with cell dimensions δx and δy, however, there are some cut cells located in the curved

boundaries which will be discussed later.

In the derivation of the two dimensional discretisation, an important tool is *Gauss'* Theorem, which relates the volume integral of the divergence of a vector field to its flux through the enclosing surface. Mathematically, given a closed volume Λ and its boundary surface S, then for any differentiable vector field f over Λ,

$$\int_A \nabla \cdot \mathbf{f} dv = \int_s \mathbf{f} \cdot \hat{\mathbf{n}} ds,$$

In this formula, n is the outward unit normal to the surface S.

3.3.2 Conservation of Horizontal Momentum

The horizontal momentum equation can be rewritten as:

$$\frac{\partial}{\partial x}\left(uu - \nu\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(-uv + \nu\frac{\partial u}{\partial y}\right) = -\frac{\partial P}{\partial x}. \tag{3.11}$$

The variable u plays here the same role as the temperature T in conduction phenomena. In this case, we also have a source term. The analysis developed previously can be applied almost directly.

We consider the finite volume scheme which is discussed before. it is assumed that the discrete velocity $u_{i+\frac{1}{2},j}$ and the point of coordinates $[(i+\frac{1}{2})\delta x, j\delta y]$ is denoted C. As before, the integration is performed on the rectangle $ne - nw - sw - se$ of size $\delta x \times \delta y$ centred around C. The result is scaled by a factor $\frac{1}{\delta x \delta y}$ (see Figure 3.1). The result is:

$$I_e + I_n - I_w - I_s = -\frac{1}{\delta x \delta y}[(p_{i+1,j} - p_{i,j})\delta y] \tag{3.12}$$

The right hand side integral is derived in the following way:

$$\int_{y_1}^{y_2}\int_{x_1}^{x_2} \frac{\partial p}{\partial x} dx dy = \int_{y_1}^{y_2} (p_{i+1,j} - p_{i,j}) dy$$
$$\approx (p_{i+1,j} - p_{i,j})\delta y.$$

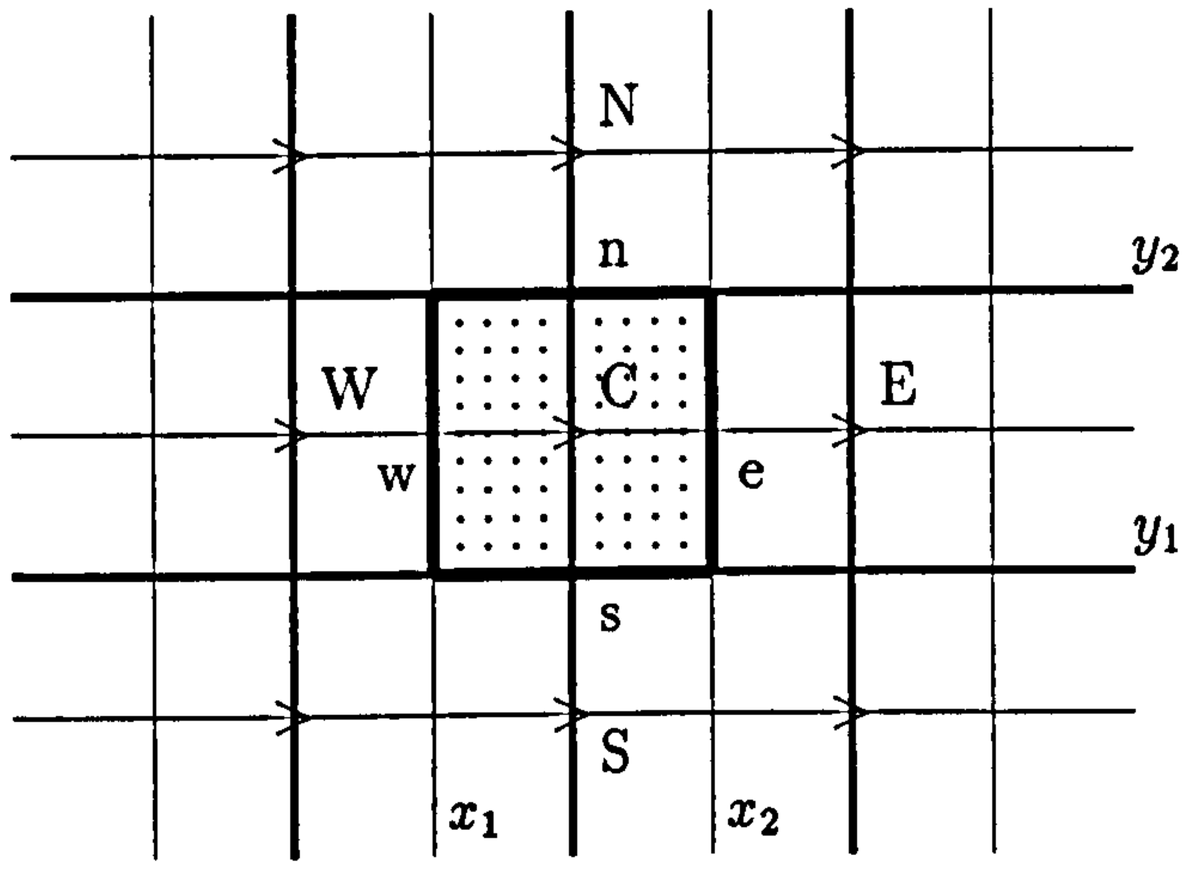


Figure 3.1: Control volume for the integration of the Navier-Stokes horizontal momentum equation

where:

$$I_e = \frac{1}{\delta x \delta y}\left[\int_{y_1}^{y_2}\left(uu - \nu\frac{\partial u}{\partial x}\right)dy\right]_{x=x_2} \tag{3.13}$$

$$I_w = \frac{1}{\delta x \delta y}\left[\int_{y_1}^{y_2}\left(uu - \nu\frac{\partial u}{\partial x}\right)dy\right]_{x=x_1} \tag{3.14}$$

$$I_n = \frac{1}{\delta x \delta y}\left[\int_{x_1}^{x_2}\left(uv - \nu\frac{\partial u}{\partial y}\right)dx\right]_{y=y_2} \tag{3.15}$$

$$I_s = \frac{1}{\delta x \delta y}\left[\int_{x_1}^{x_2}\left(uv - \nu\frac{\partial u}{\partial y}\right)dx\right]_{y=y_1} \tag{3.16}$$

I_e, I_w, I_n, and I_s are the horizontal momentum fluxes through the edges of the control volume. In order to obtain discrete equations, the integrals have to be replaced by quadrature formula, these will be approximated by central finite difference formula. For example, we have for I_e,

$$\left[\int_{y_1}^{y_2}\left(uu-\nu\frac{\partial u}{\partial x}\right)dy\right]_{x=x_2}\approx\left(u_eu_e-\nu\frac{u_E-u_C}{\delta x}\right)\delta y$$

The other integrals – I_w, I_n, and I_s are all treated in the same way. The coefficient C_i^u are defined and related to velocities but they have to be expressed in terms of the known discrete values. We have:

$$C_e^u=\frac{1}{2\delta x}u_e.$$

$$C_w^u=\frac{1}{2\delta x}u_w.$$

$$C_n^u=\frac{1}{2\delta x}u_n.$$

$$C_s^u=\frac{1}{2\delta x}u_s.$$

also

$$D_e^u=D_w^u=\frac{\nu}{\delta x^2},$$

$$D_n^u=D_s^u=\frac{\nu}{\delta y^2},$$

so that the momentum fluxes can be rewritten as following:

$$I_e=2C_e^uu_e-D_e^u(u_E-u_C)$$

$$I_w=-2C_w^uu_w-D_w^u(u_C-u_W)$$

$$I_n=2C_n^uu_n-D_n^u(u_N-u_C)$$

$$I_s=-2C_s^uu_s-D_s^u(u_C-u_S)$$

The coefficients $C_e^u, C_n^u, C_w^u, C_s^u$ involve velocities at points where the corresponding grid functions are not defined. For instance, u_e, the eastern edge velocity of the control volume is not available directly, it has to be interpolated from neighbouring values (see Figure 3.2):

$$C_e^u\approx\frac{1}{2\delta x}(\frac{u_{i+\frac{1}{2},j}+u_{i+\frac{3}{2},j}}{2})=\frac{1}{4\delta x}(u_{i+\frac{1}{2},j}+u_{i+\frac{3}{2},j}),$$

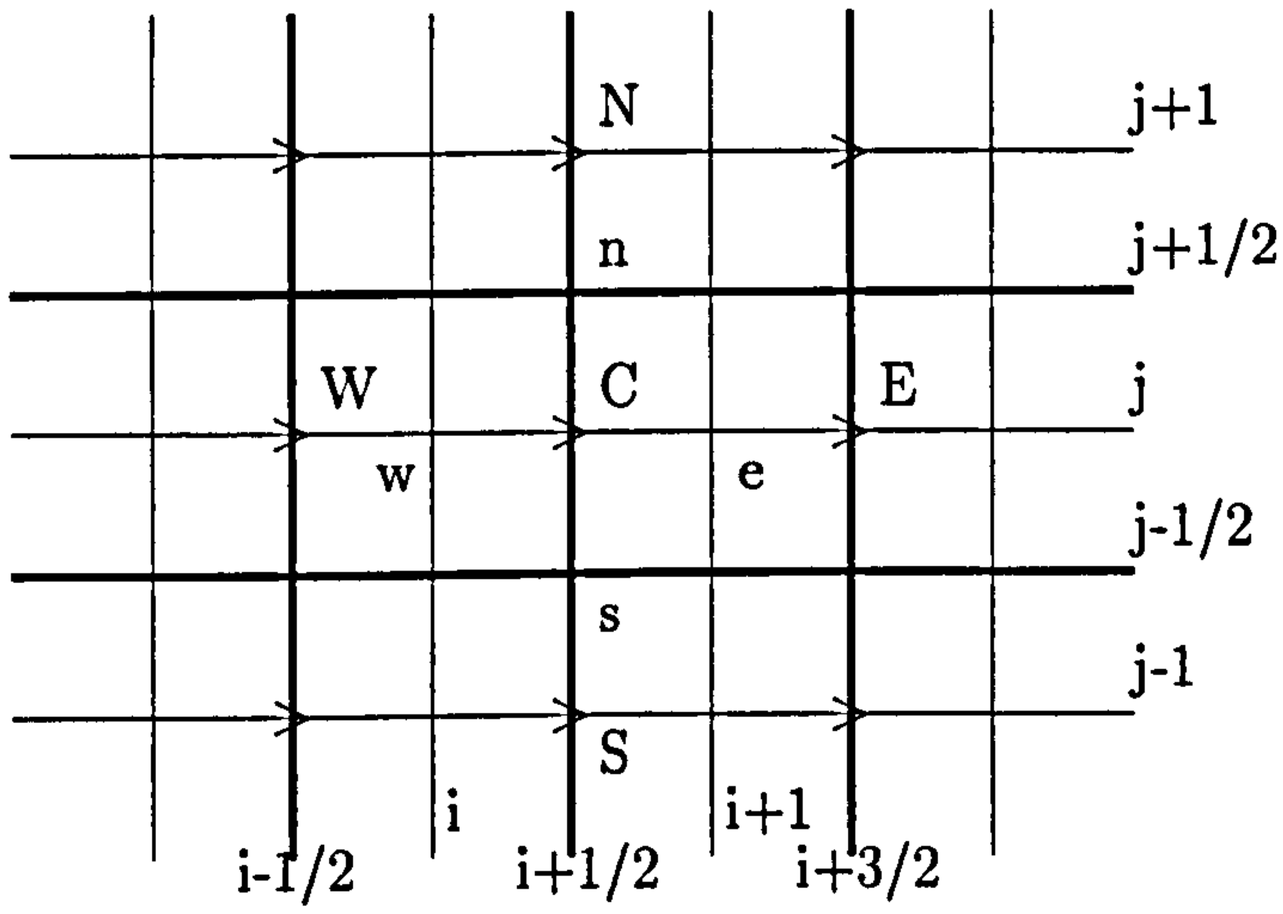


Figure 3.2: A part of the two-dimensional grid used for discretisation

As well as that, in order to have homogeneous definitions for the coefficients C_i, the signs of C_n^u, C_w^u, and C_s^u are inverted:

$$C_n^u = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + v_{i+1,j+\frac{1}{2}}),$$

$$C_w^u = \frac{1}{4\delta x}(u_{i+\frac{1}{2},j} + u_{i-\frac{1}{2},j}),$$

$$C_s^u = \frac{1}{4\delta y}(v_{i,j-\frac{1}{2}} + v_{i+1,j-\frac{1}{2}}).$$

The different finite volume schemes will be obtained for different approximations of the convective terms. In particular, different approximations of u_e, u_w, u_n, and u_s will lead to the central, upwind or hybrid schemes. Since the coefficients C_i^u and D_i^u are measures of the convective and diffusive effects respectively, the switching between a central and an upwind discretisation is done according to the relative weights.

This procedure is illustrated in the case of I_e and applies directly to the other fluxes. If $|C_e^u| < D_e^u$, central differencing is stable and u_e is integrated as:

$$u_e \approx \frac{u_E + u_C}{2}$$

The discrete flux is therefore:

$$I_e = C_e^u(u_E + u_C) - D_e^u(u_E - u_C)$$
$$= u_E(C_e^u - D_e^u) + u_C(C_e^u + D_e^u). \quad (3.17)$$

If $|C_e^u| > D_e^u$, the convective terms dominate, central differencing is unstable and needs to be replaced by upwind differencing, Since numerical diffusion introduced by the upwind dominates the physical diffusion, the latter is neglected so that only the convective flux is taken into account. Upwind differencing is achieved by an upwind interpolation of u_e. Given that $C_e^u = u_e$, its signs give the local direction of the flow and can be used to determine the direction of the upwinding:

$$u_e \approx \begin{cases} u_E, & \text{if } C_e^u < 0; \\ u_C, & \text{if } C_e^u > 0. \end{cases}$$

The discrete upwind flux is then has:

$$I_e \approx \begin{cases} 2C_e^u u_E, & \text{if } C_e^u < 0; \\ 2C_e^u u_C, & \text{if } C_e^u > 0. \end{cases}$$

$$= u_E(C_e^u - |C_e^u|) + u_C(C_e^u + |C_e^u|). \quad (3.18)$$

Combining equations(3.17) and (3.18), the hybrid discretisation of the flux I_e can be rewritten as:

$$I_e = -\frac{1}{2}[(D_e^u + |C_e^u| + |D_e^u - |C_e^u||) - C_e^u]u_E + \frac{1}{2}[D_e^u + |C_e^u| + |D_e^u - |C_e^u|| + C_e^u]u_C$$
$$= -[max(D_e^u, |C_e^u|) - C_e^u]u_E + [max(D_e^u, |C_e^u|) + C_e^u]u_C). \quad (3.19)$$

and similarly, the other fluxes are:

$$I_w = -[max(D_w^u, |C_w^u|) - C_w^u]u_W + [max(D_w^u, |C_w^u|) + C_w^u]u_C). \quad (3.20)$$

$$I_n = -[max(D_n^u, |C_n^u|) - C_n^u]u_N + [max(D_n^u, |C_n^u|) + C_n^u]u_C). \quad (3.21)$$

$$I_s = -[max(D_s^u, |C_s^u|) - C_s^u]u_S + [max(D_s^u, |C_s^u|) + C_s^u]u_C). \tag{3.22}$$

The conservation of discrete horizontal momentum equation can be written as following:

$$A_C^u u_C = A_E^u u_E + A_N^u u_N + A_W^u u_W + A_S^u u_S - \frac{1}{\delta y}(p_{i+1,j} - p_{i,j}), \tag{3.23}$$

with

$$A_C^u = A_E^u + A_N^u + A_W^u + A_S^u,$$

$$A_E^u = max(|C_e^u|, D_e^u) - C_e^u,$$

$$A_W^u = max(|C_w^u|, D_w^u) + C_w^u,$$

$$A_N^u = max(|C_n^u|, D_n^U) - C_n^u,$$

$$A_S^u = max(|C_s^u|, D_s^u) + C_s^u.$$

Under the condition that:

$$C_e^u - C_w^u + C_n^u - C_s^u = 0$$

this equation expresses the physical fact that mass should be conserved in the control volume used to derive the momentum equations. Indeed, this equation implies that:

$$\delta y u_e - \delta y u_w + \delta x u_n - \delta x u_s = 0$$

which is a discrete approximation of the mass flux through the control volume.

3.3.3 Conservation of Vertical Momentum

If the vertical momentum equation is considered a similar formula can be derived for the vertical velocities v, but this time the control volume is centred on $u_{i,j+\frac{1}{2}}$ (see Figure 3.3). The finite difference formula is:

$$A_C^v u_C = A_E^v v_E + A_N^v v_N + A_W^v v_W + A_S^v v_S - \frac{1}{\delta x}(p_{i,j+1} - p_{i,j}). \tag{3.24}$$

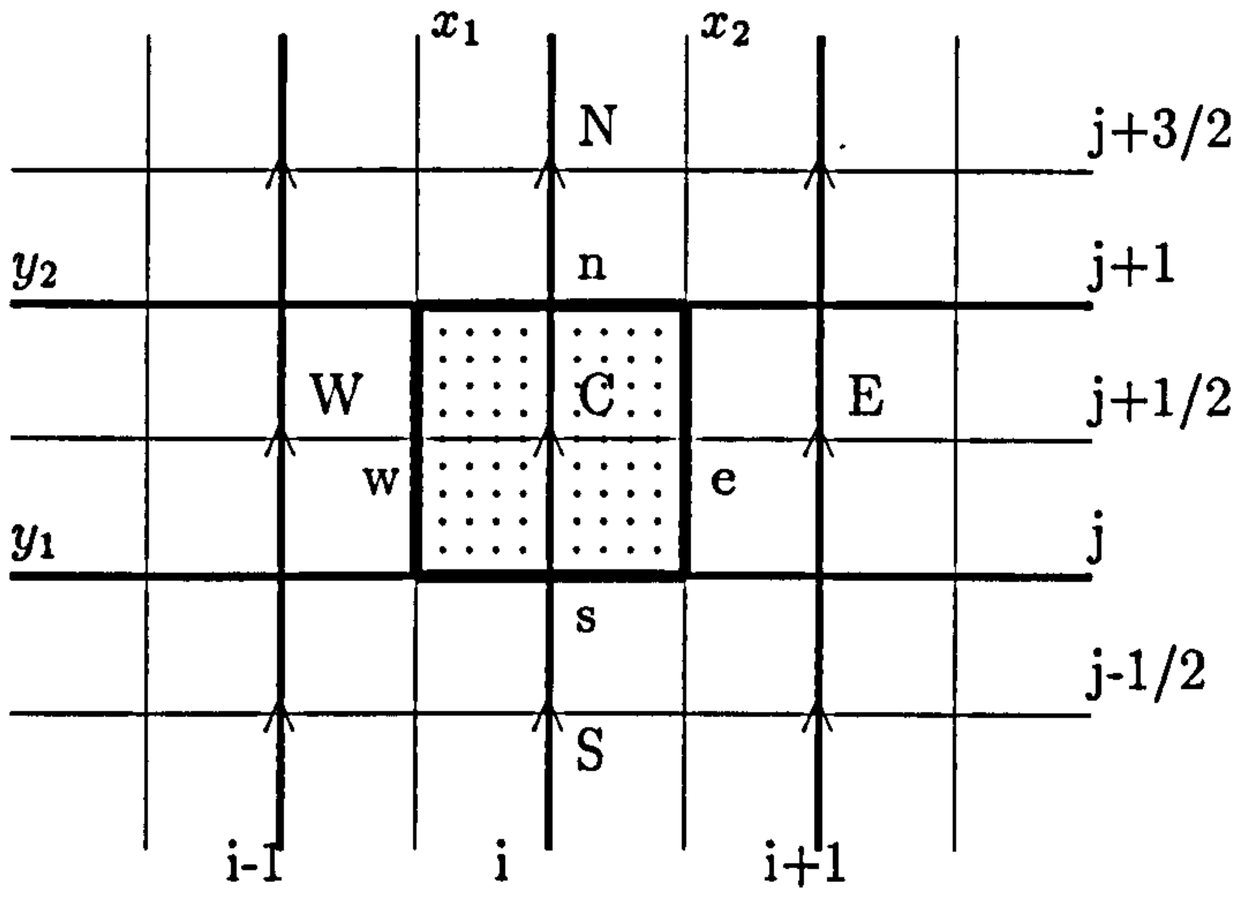


Figure 3.3: Control volume for the integration of the Navier-Stokes vertical momentum equation

The coefficients are defined in the following way:

$$A_C^v = A_E^v + A_N^v + A_W^v + A_S^v,$$

$$A_E^v = max(|C_e^v|, D_e^v) - C_e^v,$$

$$A_W^v = max(|C_w^v|, D_w^v) + C_w^v,$$

$$A_N^v = max(|C_n^v|, D_n^v) - C_n^v,$$

$$A_S^v = max(|C_s^v|, D_s^v) + C_s^v,$$

$$C_e^v = \frac{1}{4\delta x}(u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j+1}),$$

$$C_n^v = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + v_{i,j+\frac{3}{2}}),$$

$$C_w^v = \frac{1}{4\delta x}(u_{i-\frac{1}{2},j} + u_{i-\frac{1}{2},j+1}),$$

$$C_s^v = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + +v_{i,j-\frac{1}{2}}),$$

$$D_e^v = D_w^v = \frac{\nu}{\delta x^2},$$
$$D_n^v = D_s^v = \frac{\nu}{\delta y^2}.$$

3.3.4 Conservation of Mass

The derivation by integral means of the finite difference formula expressing the conservation of mass is quite straightforward. The differential form of the equation.

$$u_x + v_y = 0, \tag{3.25}$$

is integrated over one cell. The result is divided by the area of the cell for consistency again:

$$\frac{1}{\delta x \delta y} \int_{x_1}^{x_2} \int_{y_1}^{y_2} [u + v] dx dy = 0,$$
$$\frac{1}{\delta x \delta y} \int_{x_1}^{x_2} [v]_{y_1}^{y_2} + \int_{y_1}^{y_2} [u]_{x_1}^{x_2} dy = 0.$$

By approximating the integrals by

$$\delta y(u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}),$$

and

$$\delta x(v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}).$$

respectively, that is by treating the velocities as constant over each side and after division by $\delta x \delta y$, the usual form of the continuity equation is obtained:

$$\frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{\delta x} + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{\delta y} = 0. \tag{3.26}$$

3.3.5 Summary of the Results

The steady state Navier-Stokes equations for an incompressible viscous flow, discretized on a staggered grid by the finite volume method using hybrid schemes, as implemented in the PAMG code, are:

$$\frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{\delta x} + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{\delta y} = 0. \tag{3.27}$$

$$A_C^u u_{i+\frac{1}{2},j} = A_E^u u_{i+\frac{3}{2},j} + A_N^u u_{i+\frac{1}{2},j+1} + A_W^u u_{i-\frac{1}{2},j} + A_S^u u_{i+\frac{1}{2},j-1} - \frac{1}{\delta y}(p_{i+1,j} - p_{i,j}), \tag{3.28}$$

$$A_C^v v_{i+\frac{1}{2},j} = A_E^v v_{i+1,j+\frac{1}{2}} + A_N^v v_{i,j+\frac{3}{2}} + A_W^v v_{i-1,j+\frac{1}{2}} + A_S^v v_{i,j-\frac{1}{2}} - \frac{1}{\delta x}(p_{i,j+1} - p_{i,j}), \tag{3.29}$$

with

$$A_C^u = A_E^u + A_N^u + A_W^u + A_S^u, \tag{3.30}$$

$$A_E^u = max(|C_e^u|, D_e^u) - C_e^u,$$

$$A_W^u = max(|C_w^u|, D_w^u) + C_w^u,$$

$$A_N^u = max(|C_n^u|, D_n^U) - C_n^u,$$

$$A_S^u = max(|C_s^u|, D_s^u) + C_s^u,$$

$$C_e^u = \frac{1}{4\delta x}(u_{i+\frac{1}{2},j} + u_{i+\frac{3}{2},j}),$$

$$C_n^u = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + v_{i+1,j+\frac{1}{2}}),$$

$$C_w^u = \frac{1}{4\delta x}(u_{i+\frac{1}{2},j} + u_{i-\frac{1}{2},j}),$$

$$C_s^u = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + v_{i+1,j-\frac{1}{2}}),$$

$$D_e^u = D_w^u = \frac{\nu}{\delta x^2},$$

$$D_n^u = D_s^u = \frac{\nu}{\delta y^2},$$

$$A_C^v = A_E^v + A_N^v + A_W^v + A_S^v, \tag{3.31}$$

$$A_E^v = max(|C_e^v|, D_e^v) - C_e^v,$$

$$A_W^v = max(|C_w^v|, D_w^v) + C_w^v,$$

$$A_N^v = max(|C_n^v|, D_n^v) - C_n^v,$$

$$A_S^v = max(|C_s^v|, D_s^v) + C_s^v,$$

$$C_e^v = \frac{1}{4\delta x}(u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j+1}),$$

$$C_n^v = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + v_{i,j+\frac{3}{2}}),$$

$$C_w^v = \frac{1}{4\delta x}(u_{i-\frac{1}{2},j} + u_{i-\frac{1}{2},j+1}),$$

$$C_s^v = \frac{1}{4\delta y}(v_{i,j+\frac{1}{2}} + +v_{i,j-\frac{1}{2}}),$$

$$D_e^v = D_w^v = \frac{\nu}{\delta x^2},$$

$$D_n^v = D_s^v = \frac{\nu}{\delta y^2}.$$

3.4 Local Truncation Error Investigation

In the previous section, the governing equations and their discretisation are analyzed. Discrete difference equations are obtained by applying the finite volume method to the governing equations. The hybrid scheme introduced by Spading is used to derive discrete equations which is accurate. Both the use of staggered grids and hybrid schemes are design choices which are key factors for the robustness and efficiency of the solution algorithms used in PAMG and CC-PAMG. In particular, an argument to explain the accurate of the finite volume solution algorithms is addressed in the following section. It is difficulty to investigate the local truncation error with finite volume methods, so we using the finite difference formulations to analyze the truncation errors. The Taylor series expansion of the function u will be considered to investigate the accuracy problems.

A standard staggered mesh is overlaid on the domain, the velocities are associated with the cell faces and the pressure is associated with the cell centres. There are two different kinds of Taylor series expansion of the function u in the horizontal momentum equation. One is to defined the edge velocities (u_e, u_w, u_n, and u_s) of the control volume by interpolated from neighbouring values (u_E, u_W, u_N, u_S, and u_C). Then used the Taylor series expansion to investigate the local truncation error in half step ($\frac{\Delta x}{2}$). The another way is to be expanded in a Taylor series about u_C in full step (Δx). The error reduced 4 times for the finer grid according to Richardson

Extrapolation (section 4.5.3).

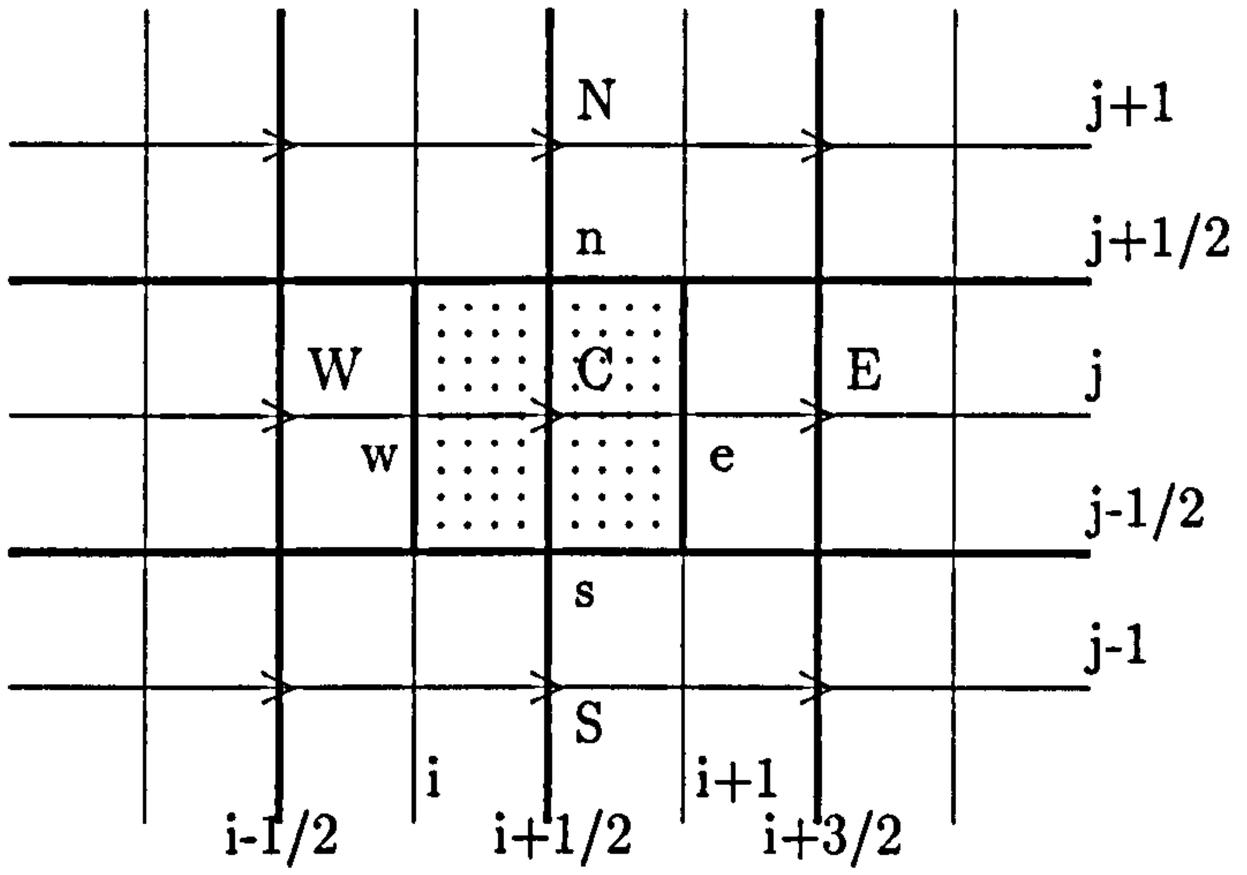


Figure 3.4: A part of the two-dimensional grid used for discretisation

The horizontal momentum equation is derived by using the central differencing and hybrid scheme, the equation can be written:

$$A_C^u u_C = A_E^u u_E + A_N^u u_N + A_W^u u_W + A_S^u u_S - \frac{1}{\delta y}(p_{i+1,j} - p_{i,j}), \qquad (3.32)$$

where the u_E, u_W, u_N, and u_S are not located on cell edge of the control volume, the edge velocities (u_e, u_w, u_n, and u_s) are interpolated from neighbouring values. For the control volume the Taylor series development of $u(x_c + \frac{\Delta x}{2}, y_c)$ about (x_c, y_c) is:

$$u(x_c + \frac{\Delta x}{2}, y_c) = u(x_c, y_c) + \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 u}{\partial x^2} + \frac{(\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 u}{\partial x^3} + \frac{(\frac{\Delta x}{2})^4}{4!}\frac{\partial^4 u}{\partial x^4} + \ldots \qquad (3.33)$$

Consider the Taylor series expansion of $u(x_c - \frac{\Delta x}{2}, y_c)$ about (x_c, y_c):

$$u(x_c - \frac{\Delta x}{2}, y_c) = u(x_c, y_c) - \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 u}{\partial x^2} - \frac{(\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 u}{\partial x^3} + \frac{(\frac{\Delta x}{2})^4}{4!}\frac{\partial^4 u}{\partial x^4} + \ldots \qquad (3.34)$$

and

$$u(x_c, y_c + \frac{\Delta y}{2}) = u(x_c, y_c) + \frac{\Delta y}{2}\frac{\partial u}{\partial y} + \frac{(\frac{\Delta y}{2})^2}{2!}\frac{\partial^2 u}{\partial y^2} + \frac{(\frac{\Delta y}{2})^3}{3!}\frac{\partial^3 u}{\partial y^3} + \frac{(\frac{\Delta y}{2})^4}{4!}\frac{\partial^4 u}{\partial y^4} + \ldots \qquad (3.35)$$

$$u(x_c, y_c - \frac{\Delta y}{2}) = u(x_c, y_c) - \frac{\Delta y}{2}\frac{\partial u}{\partial y} + \frac{(\frac{\Delta y}{2})^2}{2!}\frac{\partial^2 u}{\partial y^2} - \frac{(\frac{\Delta y}{2})^3}{3!}\frac{\partial^3 u}{\partial y^3} + \frac{(\frac{\Delta y}{2})^4}{4!}\frac{\partial^4 u}{\partial y^4} + \ldots \quad (3.36)$$

Adding Equation (3.33) and (3.34), one obtains:

$$u(x_c + \frac{\Delta x}{2}, y_c) + u(x_c - \frac{\Delta x}{2}, y_c) = 2u(x_c, y_c) + 2\frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 u}{\partial x^2} + 2\frac{(\frac{\Delta x}{2})^4}{4!}\frac{\partial^4 u}{\partial x^4} + \ldots \quad (3.37)$$

Subtracting Equation (3.34) and (3.33), one obtains:

$$u(x_c + \frac{\Delta x}{2}, y_c) - u(x_c - \frac{\Delta x}{2}, y_c) = 2\frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 u}{\partial x^3} + \ldots \quad (3.38)$$

With the same way, we can have:

$$u(x_c, y_c + \frac{\Delta y}{2}) + u(x_c, y_c - \frac{\Delta y}{2}) = 2u(x_c, y_c) + 2\frac{(\frac{\Delta y}{2})^2}{2!}\frac{\partial^2 u}{\partial y^2} + 2\frac{(\frac{\Delta y}{2})^4}{4!}\frac{\partial^4 u}{\partial y^4} + \ldots \quad (3.39)$$

$$u(x_c, y_c + \frac{\Delta y}{2}) - u(x_c, y_c - \frac{\Delta y}{2}) = 2\frac{\Delta y}{2}\frac{\partial u}{\partial y} + \frac{(\frac{\Delta y}{2})^3}{3!}\frac{\partial^3 u}{\partial y^3} + \ldots \quad (3.40)$$

Solving for $\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial y}$, $\frac{\partial^2 u}{\partial x^2}$, and $\frac{\partial^2 u}{\partial y^2}$,

$$\frac{\partial u}{\partial x} = \frac{u(x_c + \frac{\Delta x}{2}, y_c) - u(x_c - \frac{\Delta x}{2}, y_c)}{\frac{\Delta x}{2}} + O(\frac{\Delta x}{2})^2 \quad (3.41)$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{u(x_c + \frac{\Delta x}{2}, y_c) - 2u(x_c, y_c) + u(x_c - \frac{\Delta x}{2}, y_c)}{(\frac{\Delta x}{2})^2} + O(\frac{\Delta x}{2})^2 \quad (3.42)$$

$$\frac{\partial u}{\partial y} = \frac{u(x_c, y_c + \frac{\Delta y}{2}) - u(x_c, y_c - \frac{\Delta y}{2})}{\frac{\Delta y}{2}} + O(\frac{\Delta y}{2})^2 \quad (3.43)$$

$$\frac{\partial^2 u}{\partial y^2} = \frac{u(x_c, y_c + \frac{\Delta y}{2}) - 2u(x_c, y_c) + u(x_c, y_c - \frac{\Delta y}{2})}{(\frac{\Delta y}{2})^2} + O(\frac{\Delta y}{2})^2 \quad (3.44)$$

For the pressure coefficients ($p_{i,j}$ and $p_{i+1,j}$), we have:

$$p_{i,j} = p(x_c - \frac{\Delta x}{2}, y_c) = p(x_c, y_c) - \frac{\Delta x}{2}\frac{\partial p}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 p}{\partial x^2} - \frac{((\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 p}{\partial x^3} + \ldots$$

$$p_{i+1,j} = p(x_c + \frac{\Delta x}{2}, y_c) = p(x_c, y_c) + \frac{\Delta x}{2}\frac{\partial p}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 p}{\partial x^2} + \frac{((\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 p}{\partial x^3} + \ldots$$

Solving for $\frac{\partial p}{\partial x}$, i. e.,

$$\frac{\partial p}{\partial x} = \frac{p(x_c + \frac{\Delta x}{2}, y_c) - p(x_c - \frac{\Delta x}{2}, y_c)}{\frac{\Delta x}{2}} + O(\frac{\Delta x}{2})^2 \quad (3.45)$$

Thus, a second-order accurate finite difference approximation for horizontal momentum equation has been obtained. A similar expression for vertical momentum equation can be generated by replacing the horizontal velocity with the vertical velocity, then using the Taylor series expansing method. It has the same accurate as the horizontal momentum equation $\{O[(\frac{\Delta x}{2})^2, (\frac{\Delta y}{2})^2]\}$.

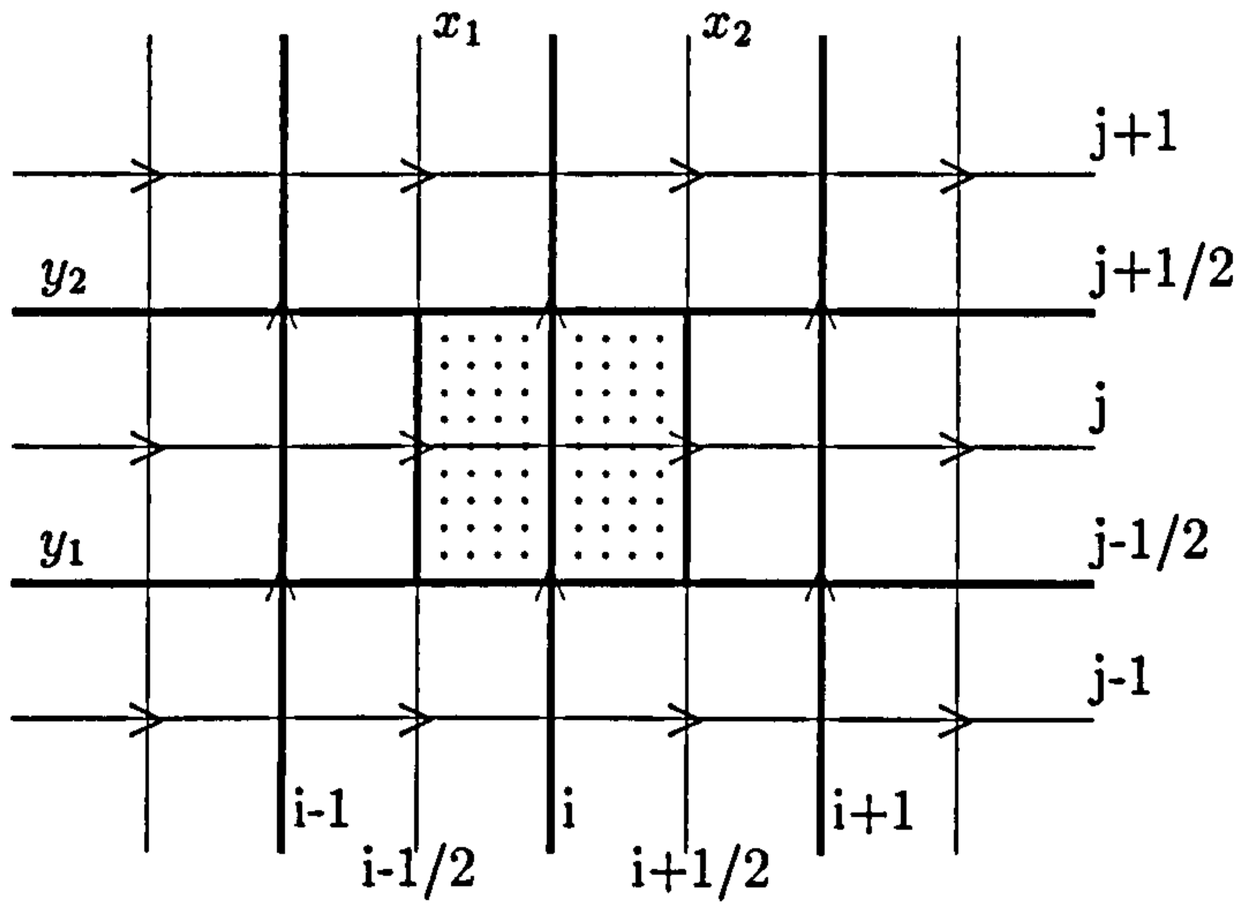


Figure 3.5: Control volume for the integration of the Navier-Stokes continuity equation

For the Navier-Stokes continuity equation, the Taylor series expansion about the (x_i, y_j) is:

$$u(x_i + \frac{\Delta x}{2}, y_j) = u(x_i, y_j) + \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 u}{\partial x^2} + \frac{(\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 u}{\partial x^3} + \frac{(\frac{\Delta x}{2})^4}{4!}\frac{\partial^4 u}{\partial x^4} + \ldots \quad (3.46)$$

Consider the Taylor series expansion of $u(x_i - \frac{\Delta x}{2}, y_j)$ about (x_i, y_j):

$$u(x_i - \frac{\Delta x}{2}, y_j) = u(x_i, y_j) - \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\frac{\Delta x}{2})^2}{2!}\frac{\partial^2 u}{\partial x^2} - \frac{(\frac{\Delta x}{2})^3}{3!}\frac{\partial^3 u}{\partial x^3} + \frac{(\frac{\Delta x}{2})^4}{4!}\frac{\partial^4 u}{\partial x^4} + \ldots \quad (3.47)$$

and

$$u(x_i, y_j + \frac{\Delta y}{2}) = u(x_i, y_j) + \frac{\Delta y}{2}\frac{\partial u}{\partial y} + \frac{(\frac{\Delta y}{2})^2}{2!}\frac{\partial^2 u}{\partial y^2} + \frac{(\frac{\Delta y}{2})^3}{3!}\frac{\partial^3 u}{\partial y^3} + \frac{(\frac{\Delta y}{2})^4}{4!}\frac{\partial^4 u}{\partial y^4} + \ldots \quad (3.48)$$

$$u(x_i, y_j - \frac{\Delta y}{2}) = u(x_i, y_j) - \frac{\Delta y}{2}\frac{\partial u}{\partial y} + \frac{(\frac{\Delta y}{2})^2}{2!}\frac{\partial^2 u}{\partial y^2} - \frac{(\frac{\Delta y}{2})^3}{3!}\frac{\partial^3 u}{\partial y^3} + \frac{(\frac{\Delta y}{2})^4}{4!}\frac{\partial^4 u}{\partial y^4} + \ldots \quad (3.49)$$

Now computing the first derivative of $u(x, y)$, we have:

$$\frac{\partial u}{\partial x} = \frac{u(x_i + \frac{\Delta x}{2}, y_j) - u(x_i - \frac{\Delta x}{2}, y_j)}{\frac{\Delta x}{2}} + O(\frac{\Delta x}{2})^2 \quad (3.50)$$

$$\frac{\partial u}{\partial y} = \frac{u(x_i, y_j + \frac{\Delta y}{2}) - u(x_i, y_j - \frac{\Delta y}{2})}{\frac{\Delta y}{2}} + O(\frac{\Delta y}{2})^2 \quad (3.51)$$

which are identical to the second-order accurate expression obtained from Taylor series expansion. From this investigation, we observe that the discretisation of the Navier-Stokes equations is second-order accuracy.

3.5 Grid Transformation

Although we do not used transformation a PDE in Cartesian grid algorithm, the CFX 4.2 still use this method to create the boundary fitted structured grid which using multi-block structure. Define the following relations between the physical and computational spaces:

$$\xi = \xi(x, y) \quad (3.52)$$

$$\eta = \eta(x, y) \quad (3.53)$$

The chain rule for partial differentiation yields the following expression:

$$\frac{\partial}{\partial x} = \frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi} + \frac{\partial \eta}{\partial x}\frac{\partial}{\partial \eta} \quad (3.54)$$

the partial derivatives will be denoted using the subscripts notation, i. e., $\frac{\partial \eta}{\partial x} = \xi_x$, so,

$$\frac{\partial}{\partial x} = \xi_x\frac{\partial}{\partial \xi} + \eta_x\frac{\partial}{\partial \eta} \quad (3.55)$$

and similarly,

$$\frac{\partial}{\partial y} = \xi_y\frac{\partial}{\partial \xi} + \eta_y\frac{\partial}{\partial \eta} \quad (3.56)$$

Now consider the continuity equation,such as:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{3.57}$$

The equation may be transformed from physical space to computational space using Equation (3.55) and (3.56). As a result,

$$\xi_x \frac{\partial u}{\partial \xi} + \eta_x \frac{\partial u}{\partial \eta} + \xi_y \frac{\partial v}{\partial \xi} + \eta_y \frac{\partial v}{\partial \eta} = 0 \tag{3.58}$$

Comparing the original conservation of the mass equation and the transformed equation given in above equation, it is obvious that the transformed equation is more complicated than the original equation. Generally that is always the case. Thus, a trade-off is introduced whereby advantages gained by using the generalized coordinates are somehow counterbalanced by the resultant complexity of the of the PDE.

These transformation derivatives (ξ_x, ξ_y, η_x, and η_y) are defined as the metrics of transformation or simply as the metrics. The interpretation of the metrics is obvious considering the following approximation:

$$\xi_x = \frac{\partial \xi}{\partial x} \cong \frac{\Delta \xi}{\Delta x} \tag{3.59}$$

This expression indicates that the metrics represent the ration of arc lengths in the computational space to that of the physical space. The computational of the metrics is considered next:

$$d\xi = \xi_x dx + \xi_y dy$$

$$d\eta = \eta_x dx + \eta_y dy$$

which are rewritten in a compact form as:

$$\begin{bmatrix} d\xi \\ d\eta \end{bmatrix} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \begin{bmatrix} dx \\ dy \end{bmatrix} \tag{3.60}$$

Reversing the role of independent variables, i. e.,

$$x = x(\xi, \eta)$$

$$y = y(\xi, \eta)$$

The following differential expression are:

$$dx = x_\xi d\xi + x_\eta d\eta$$

$$dy = y_\xi d\xi + y_\eta d\eta$$

In the compact form they are rewritten as:

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \begin{bmatrix} d\xi \\ d\eta \end{bmatrix} \tag{3.61}$$

Solving Equation (3.61) for the right-hand column matrix, multiplying by the inverse of the 2×2 coefficient matrix, we have

$$\begin{bmatrix} d\xi \\ d\eta \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}^{-1} \begin{bmatrix} dx \\ dy \end{bmatrix} \tag{3.62}$$

Comparing Equation (3.60) and Equation (3.62), it can be concluded that:

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}^{-1} \tag{3.63}$$

Following the standard rules for creating the inverse of a matrix, Equation (3.63) is written as

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \frac{\begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix}}{\begin{vmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{vmatrix}} \tag{3.64}$$

Consider the determinant in the denominator of Equation (3.64). Since the value of a determinant is unchanged by transposing its terms, we have

$$\begin{vmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{vmatrix} = \begin{vmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{vmatrix} = J \tag{3.65}$$

Note that the right-hand determinant of Equation (3.65) is precisely the Jacobian J of the transformation. Substituting Equation (3.65) into Equation (3.64), one has

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix} \tag{3.66}$$

Comparing like elements of the two matrices in Equation (3.66), the relationships for the direct metrics in terms of the inverse metrics, namely,

$$\xi_x = J^{-1} y_\eta \qquad \xi_y = -J^{-1} x_\eta$$

$$\eta_x = -J^{-1} y_\xi \qquad \eta_y = J^{-1} x_\xi$$

and is defined as the Jacobian of transformation. For grid generation methods where analytical expressions for the metrics can be written, they may be analytically evaluated or determined numerically by the use of finite difference expressions which is expensive in storage and computing time.

3.6 Summary

In this chapter, we have tied together the fundamental aspects of numerical discretisation and put them together to form various techniques for the numerical solution of the continuity and momentum equations. We have seen that the choice of an appropriate numerical technique is closely related to the mathematical behaviour of the original partial differential equations. Appropriate discrete equations have been derived for the solver PAMG (and also used in CC-PAMG). The discretisation process relies on the use of staggered grids together with hybrid differencing to provide a set of non-linear algebraic equations which demonstrate the second order accuracy associated with central differencing and finite volume method. It allows easy enforcement of conservation, particularly for mass. It also leads to a very robust solver when used in conjunction with a multigrid procedure because they implicity define a expand (numerical diffusion) method.

Discretised equations of the Navier-Stokes equations must be established at the nodal point in order to solve a problem. For control volumes that are adjacent to the domain boundaries, the general discretised equation is modified to incorporate the boundary conditions. The resulting system of algebraic equations is then solved to obtain the distribution of the flows at nodal points. Once discrete equations and

boundary conditions are available, it is possible to design an algorithm to solve them. This is the subject of the next chapter.

Chapter 4

The Cartesian Cut-cell Method

4.1 Introduction

In this chapter we describe an alternative approach for dealing with complex two-dimensional geometries, the so-called Cartesian cut-cell method. Conceptually, this method is quite simple. Solid boundaries blank out areas of a background Cartesian mesh, and the resultant cut cells receive special attention during the integration of the flow solution. However, this rather simple concept belies the obstacles that must be overcome in order to achieve a practical scheme. Whilst these obstacles are far from insurmountable, they are often perceived as stumbling blocks, hence the dearth of schemes which employ the Cartesian cut method.

Our procedure for determining the required cell-type information starts by tracing the outline of each body so as to find all the intersections between the grid and the specified input geometry. These intersections are then collated to find the types and locations of all the cut cells. Given this information, it is then a simple matter to scan the mesh, thereby determining which of the remaining cells are solid (excluded), and which are un-cut. Since only cut cells are examined in detail, this method proves fairly efficient. Once the cell-type information has been gathered, a finite volume scheme has been used to integrate the discretized flow equations. The adaptive multigrid method has been used to calculate the fluid problems considered here.

4.1.1 Four Main Types Of Cartesian Cut Cells

In order to represent arbitrarily complex geometries which in general may be stationary or moving relative to a reference point, we used a Cartesian cut cell approach. Solid bodies are simply cut out of a background Cartesian mesh and their boundaries represented by different types of cut cell. Essentially, a two-dimensional Cartesian cut-cell mesh can be generated as follows:

1. Construction of a background Cartesian mesh. All mesh cells are initially flagged as inlet cells, outlet cells, or walls in the computational domain.

2. Locating the intersection points between the Cartesian mesh lines and the boundaries of solid bodies. The cells partially cut by the boundaries are registered as cut cells. Accordingly, the volume and other geometric information for each cut-cell is determined.

3. Locating solid cells. Sweeps across the background mesh are then performed to identify which cells or rows of cells are bounded by solid or cut cells. These are registered as solid cells. The new Cartesian boundary geometries are set up.

In essence, the algorithm described here is a method whereby solid wall boundary conditions may be applied on a Cartesian mesh for arbitrarily shaped bodies. Since the algorithm follows a finite volume methodology, it is not restricted to a particular system of equations, nor is it tailored to a specific numerical scheme.

There are eight main sub-types of cut cells (see Figure 4.1) which have been used in this project. The angle α formed the straight boundary line with the x-axis in

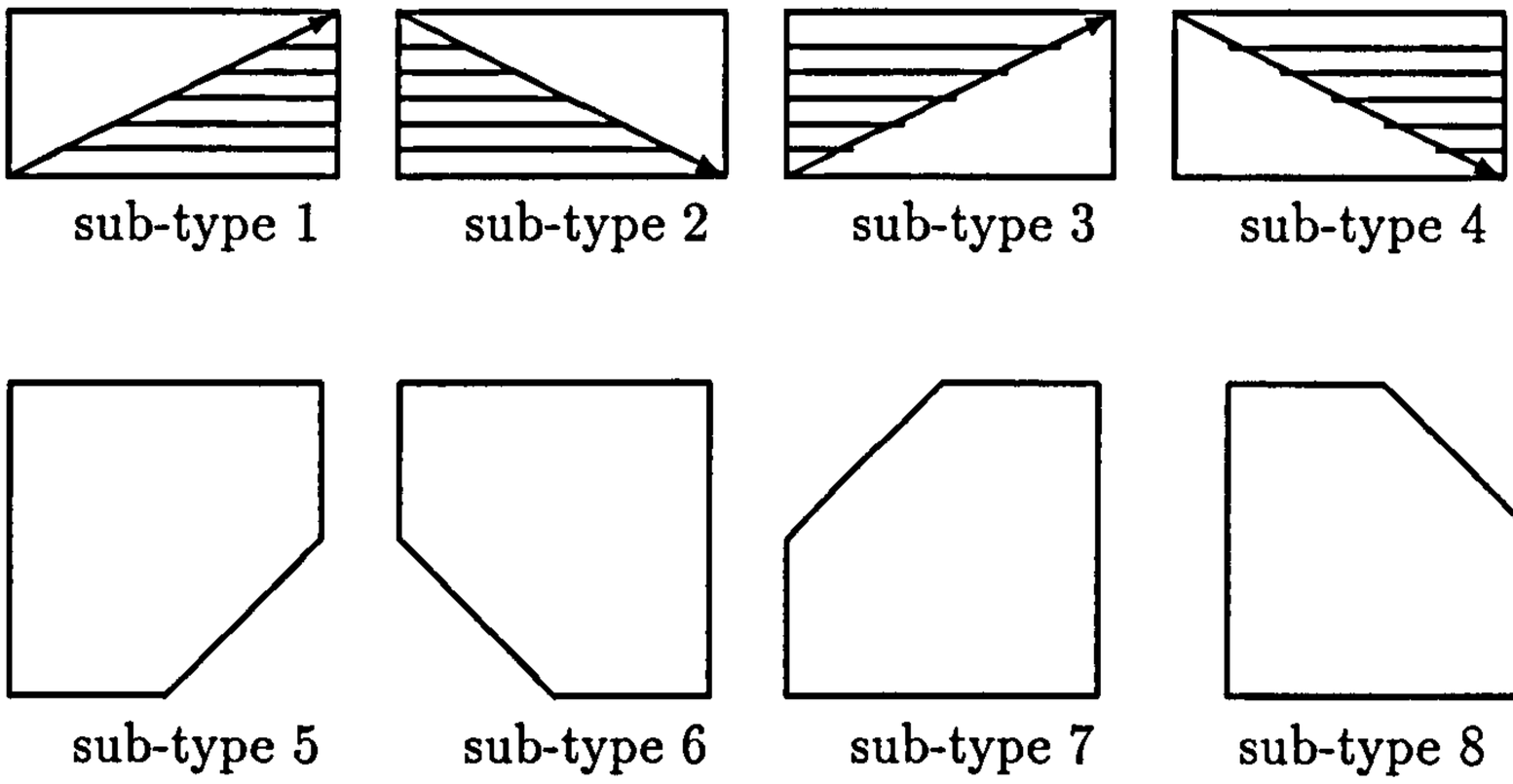


Figure 4.1: Four basic sub-types of Cartesian Cut-cell.

sub-type 1 and 5 located between 0^0 – 90^0. Whilst the angle α formed the boundary line with the x-axis lies between 180^0 – 270^0 for sub-type 3 and 7, it is found that the boundary line locates in nearly the same position but the flowfield and the solid body are located on the reverse side. Sub-types 2 and 4 have the same boundary position; again, their flowfield and the solid bodies are located on different sides as the type 1 and type 3. Sub-type 6 and 8 have the similar boundary positions whlist the cut parts are relatively smaller.

In practice, the acceptable cut-cell volume could be as large as one half of the flow cell size. For the next step, i. e. refinement, the cut-cell can only spawn three quadrants in the CC-PAMG code if half of its volume has been cut out. The coarse grid has been cut at least one quadrant for the refinement procedure.

4.2 Method For Distinguishing Boundary Meshes

Because the computational domain is established by the unfitted Cartesian cell algorithm, the physical boundaries do not correspond exactly to the boundaries of the computational domain. The first calculational step is to find the real wall boundaries.

To determine the location of a Cartesian cut-cell from the computational domain, the best way is to find that part of the cell which is located outside the flowfield. If the middle point of a cell is located on the boundaries or outside of the flowfield, that means this cell is a cut-cell and should cut out the outside part of the cell located outside the computational domain. In the refinement process, this cut-cell will spawn three children: one normal Cartesian cell and two Cartesian cut cells. There is a simple method that can hand out this problems in analytical geometry. In two-dimensional domains, we divide the domain into two parts with a line. It is easy to determine a point that is located in which side according to the relation of the point and the straight line. Because it has a different sign if it is located into the different side.

For two-dimensional problem, we need check the relative positions of a cell centre with 2 straight lines (tangent lines for the curved boundary conditions). If a cell centre is located between the two boundary lines (or tangent lines), the values of the two different boundary line equations for the centre point (x_i, y_j) will have opposite signs. If the cell centre is located above this two lines, the values of the two line equations are both positive. Otherwise, if the cell centre is posited below the two lines, they have the negative values. More details can be read in the following two sections.

4.2.1 The Relative positions of a straight line and a point

Suppose we have in the xy-plane, a point $A'(x', y')$ and a straight line L (defined by Figure 4.2):

$$ax + by + c = 0 \tag{4.1}$$

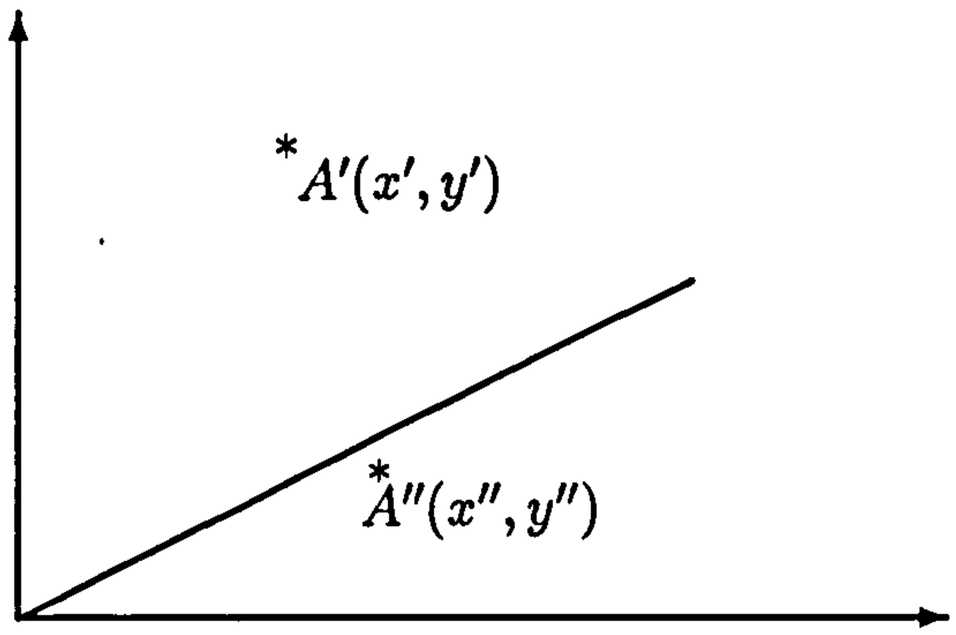


Figure 4.2: Position of the point and the straight line.

If the point A' lies on the line L, then we have:

$$ax' + by' + c = 0 \tag{4.2}$$

Let us find out what geometrical meaning the following expression has

$$h(x', y') = ax' + by' + c,$$

if the point A' is not on the straight line.

Let there be given two different points on the xy-plane: $A'(x', y')$ and $A''(x'', y'')$ which do not lie on the line L. Find the coordinates x and y of the point A which divides the segment $A'A''$ in the ratio $\lambda_1 : \lambda_2$. The coordinates of any point of the segment $A'A''$ can be represented in the form

$$x = tx' + (1-t)x'', \quad y = ty' + (1-t)y'',$$

where $t = \frac{\lambda_1}{\lambda_1+\lambda_2}$, then $1 - t = \frac{\lambda_2}{\lambda_1+\lambda_2}$, $(0 \leq \mathrm{t} \leq 1)$.

Thus for any point A of the segment $A'A''$

$$h(x, y) = th(x', y') + (1-t)h(x'', y'') = h(t) \tag{4.3}$$

If the points A' and A'' belong to one side of the straight line, then h(t) does not vanish on the interval [0,1], consequently, h(0) = $h(x', y')$ and h(1)= $h(x'', y'')$ are of

the same sign. If A' and A'' belong to different side of the line L, then h(t) vanishes on the interval [0,1] and, being a linear function, attains at the end-points values of opposite signs, i.e. $h(x', y')$ and $h(x'', y'')$ have opposite signs.

So the linear expression

$$ax' + by' + c = 0 \tag{4.4}$$

is positive for the point A' is positioned to one side of the straight line L and is negative for the points of the other. It can be used to distinguish the meshes which belongs to the calculation domain or outside the flowfield.

4.3 Method For Distinguishing Curved Boundary Meshes

For arbitrarily complex boundary conditions, the boundary geometries are no longer the combinations of straight lines, rather they must have some second-order curves in its boundary conditions. We can generalize this by considering general second-order curves. The second-order curves include the ellipse, hyperbola, parabola, and circle etc. A curve of the second-order is defined as the locus of point in the plane whose coordinates satisfy an equation of the form:

$$a_{11}x^2 + 2a_{12}xy + a_{22}y^2 + 2a_1x + 2a_2y + a = 0 \tag{4.5}$$

in which at least one of the coefficients a_{11}, a_{12}, a_{22} is non-zero.

Let us find out the geometrical meaning of second-order curves. We put the curve in the new coordinate system $x'y'$ which is related to the xy -system by the formulas,

$$x = x' \cos\alpha + y' \sin\alpha$$

$$y = x' \sin\alpha + y' \cos\alpha$$

The equation of the curve, preserving the form given by Equation (4.5), will have in the $x'y'$ -system the coefficient

$$2a_1' = 2a_{11}\cos\alpha\sin\alpha - 2a_{22}\sin\alpha\cos\alpha + 2a_{12}(\cos^2\alpha - \sin^2\alpha)$$

$$= (a_{11} - a_{22})\sin 2\alpha + 2a_{12}\cos 2\alpha$$

Obviously, it is always possible to choose the angle α so that this coefficient is equal to zero. Therefore, without loss of generality, we may set $a_{12} = 0$ in the initial Equation (4.5) . By passing over to the new coordinate system $x'y'$,

$$x' = x + \frac{a_1}{a_{11}}, \qquad y' = y + \frac{a_2}{a_{22}}$$

We cast the Equation (4.5) in the form:

$$a_{11}x'^2 + a_{22}y'^2 + c = 0 \tag{4.6}$$

and consider the following sub-cases:

A1: c $\neq$ 0; a_{11} and a_{22} are of the same sign which is opposite to the sign of c. The curve is obviously an ellipse.

A2: c $\neq$ 0; a_{11} and a_{22} have different signs. The curve is a hyperbola.

A3: c $\neq$ 0; a_{11}, a_{22} , and c have the same sign. None of the real points satisfies the equation. The curve is called imaginary.

A4: c = 0; a_{11} and a_{22} have different signs. The curve decomposes into two straight lines, since Equation (4.6) can be written in the form:

$$\left(x' - \sqrt{-\frac{a_{22}}{a_{11}}y'}\right)\left(x' + \sqrt{-\frac{a_{22}}{a_{11}}y'}\right) = 0$$

A5: c = 0; a_{11} and a_{22} have the same sign. The equation can be written in the form:

$$\left(x' - i\sqrt{-\frac{a_{22}}{a_{11}}y'}\right)\left(x' + i\sqrt{-\frac{a_{22}}{a_{11}}y'}\right) = 0$$

The curve decomposes into a pair of imaginary straight lines intersecting at a real point(0,0). Thus, a real curve of the second order represents either a conic section (the ellipse, hyperbola, parabola), or a pair of straight lines (which may even coincide).

4.3.1 A Tangent Line to a Conic Section

To determine the location of a Cartesian cut-cell from a conic section boundary, the best way is to find that portion of the cell that is located outside the curved flowfield. We can use the tangent line of the curve to check that the centre of the grid lies in or out of the flowfield (see Figure 4.3).

The tangent line to a curve at point A is defined as the limiting position, if this exists, of the secant line AB when the point B approaches A unboundedly. Suppose a curve is given by the equation $y = f(x)$. Let us form the equation of a tangent line at a point $a(x_0, y_0)$. Let $B(x_0 + \Delta x, y_0 + \Delta y)$ be a point of the curve situated close to A. The equation of the secant is:

$$y - y_0 = \frac{\Delta y}{\Delta x}(x - x_0)$$

as $B \longrightarrow A$

$$\frac{\Delta y}{\Delta x} \longrightarrow f'(x_0)$$

and we get the equation of the tangent line:

$$y - y_0 = f'(x_0)(x - x_0) \tag{4.7}$$

Analogously, if a curve is specified by the equation $x = \varphi(y)$, then the equation of the tangent line at point (x_0, y_0) will be:

$$x - x_0 = \varphi'(y_0)(y - y_0) \tag{4.8}$$

Now let us form the equation of a tangent line to a conic section. The equation of the parabola may be written in the form:

$$x = \frac{y^2}{2p}$$

then the equation of the tangent line in equation (4.8) will be:

$$x - x_0 = \frac{y_0}{p}(y - y_0),$$

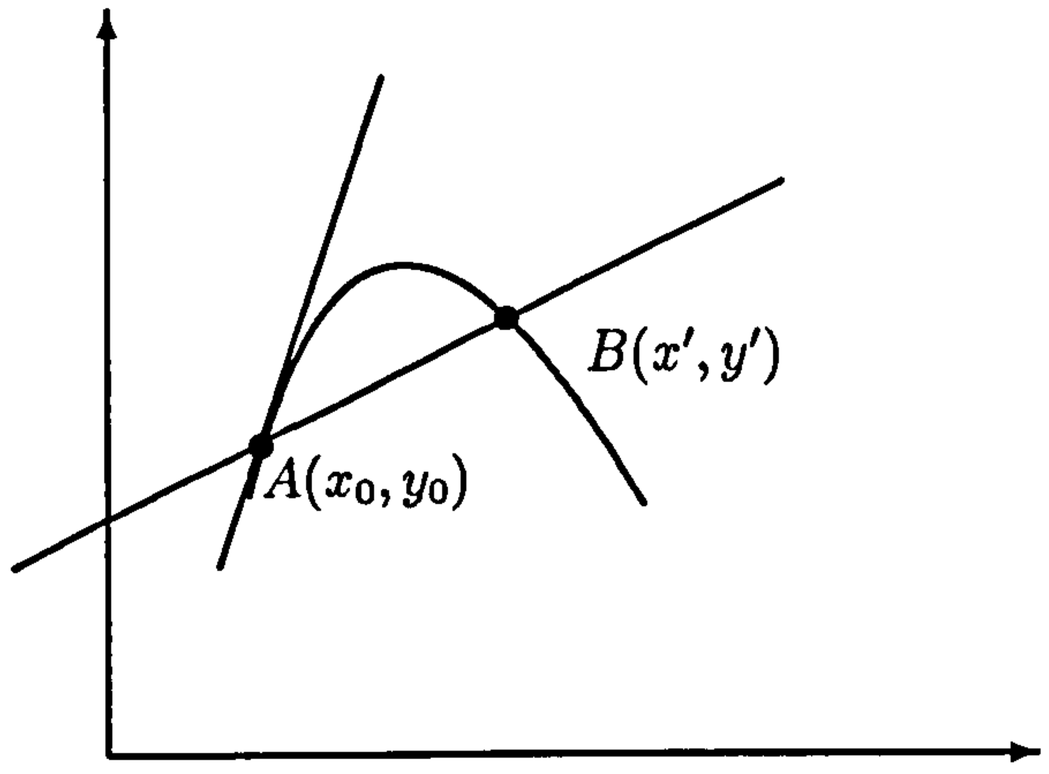


Figure 4.3: A tangent line to a conic section.

or

$$yy_0 - {y_0}^2 + px_0 - px = 0.$$

Since the point (x_0, y_0) lies on the parabola and $y_0^2 - 2px_0 = 0$, the equation of the tangent line can be represented in the following final form:

$$yy_0 - p(x - x_0) = 0. \tag{4.9}$$

Let us form the ellipse (hyperbola) equation of a tangent line, where (x_0, y_0) is a point on the ellipse, and $y_0 \neq 0$. In the neighbourhood of this point the ellipse can be specified by the equation:

$$y = b\sqrt{1 - \frac{x^2}{a^2}},$$

where the square root should be taken with the same sign as y_0. The equation of the tangent line is found by the Equation (4.7):

$$y - y_0 = -\frac{x_0 b}{a^2\sqrt{1 - \frac{x_0^2}{a^2}}}(x - x_0),$$

or

$$y - y_0 = -\frac{x_0 b^2}{y_0 a^2}(x - x_0).$$

Multiplying by y_0/b^2 and transposing all terms to the left-hand side, we obtain:

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} - (\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2}) = 0,$$

or

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} - 1 = 0,$$

since

$$\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2} = 1.$$

In the neighbourhood of any point (x_0, y_0) of the ellipse, where $x_0 \neq 0$, the ellipse can be specified by the equation:

$$x = a\sqrt{1 - \frac{y^2}{b^2}}$$

The square root is taken with the same sign as x_0. Then, reasoning in a similar way and using the Equation (4.8), we arrive to the equation of the tangent line

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} = 1 \tag{4.10}$$

Since at each point of the ellipse x_0 and y_0 cannot both be equal to zero, then at any point (x_0, y_0) the equation of the tangent line to the ellipse will be

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} = 1.$$

The equation of the tangent line to the hyperbola

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

is obtained analogously and has the form

$$\frac{xx_0}{a^2} - \frac{yy_0}{b^2} = 1$$

Let us show that a tangent line to a conic section has only one point in common with this section (i.e. the point of tangency). Indeed, let us take an ellipse whose equation is:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

The equation of the tangent line at point (x_0, y_0) will be:

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} = 1$$

Let us now look for the points of intersection of the ellipse with its tangent line. Eliminating x from the equations we obtain for y :

$$\frac{y^2}{b^2} + \frac{a^2}{x_0^2}(\frac{yy_0}{b^2} - 1)^2 = 0,$$

or

$$y^2 \frac{a^2}{b^2 x_0^2}\left(\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2}\right) - 2y\frac{a_0^2}{x_0^2}\frac{y_0}{b^2} + \frac{a^2}{x_0^2}\left(1 - \frac{x_0^2}{a^2}\right) = 0.$$

Since the point (x_0, y_0) lies on the ellipse, we have $\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2} = 1$, and the equation for y takes the form:

$$\frac{a^2}{b^2 x_0^2}(y^2 - 2yy_0 + y_0^2) = 0$$

This equation has two merged roots $y = y_0$. Analogously, eliminating y from the equations of the ellipse and its tangent line we obtain $x = x_0$. Thus, the ellipse has only one point in common with the tangent line, i.e. the point of tangency (x_0, y_0). For the hyperbola and parabola this is proved in a similar way.

Based on the property of a tangent line to have only one point in common with a conic section is a refined method of deriving the equations of a pair of tangent lines passing through an arbitrary point. Let us takes for example, an ellipse specified by the following equation:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1.$$

We then form the equation of the tangent lines to the ellipse passing through the point (x_0, y_0) not lying on the ellipse. Let (x, y) be an arbitrary point. The coordinates of any point (x', y') on the straight line g passing through the points (x_0, y_0) and (x, y) can be represented in the form

$$x' = \frac{x_0 + tx}{1 + t}$$

Procedure $Bdry\_check(x_j, y_j, Value_T, Value_B)$
Real: $x_{I_1}, y_{I_1}, x_{II_1}, y_{II_1}$!Start positions
Real: $x_{I_2}, y_{I_2}, x_{II_2}, y_{II_2}$!end positions
If ($x_{I_1}, \leq x_i \leq x_{I_2}$ or $x_{II_1} \leq x_i \leq x_{II_2}$) then
If ($y_{I_1}, \leq y_i \leq y_{I_2}$ or $y_{II_1} \leq y_i \leq y_{II_2}$) then
Calculate y_{I_0} ! at $x_{I_0} = x_i$
Calculate y_{II_0} ! at $x_{II_0} = x_i$
Top tangent line equation !$at(x_{I_0}, y_{I_0})$
Bottom tangent line equation !$at(x_{II_0}, y_{II_0})$
$Value_T$ = Top tangent line equation !$x = x_i, y = y_j$
$Value_B$ = Bottom tangent line equation !$x = x_i, y = y_j$
End if
End if
End procedure

Figure 4.4: Pseudo-code procedure to distinguish the boundary meshes.

$$y' = \frac{y_0 + ty}{1 + t}$$

We now look for the points of intersection of the line g with the ellipse, for which purpose substituting x' and y' in the equation of the ellipse. We get:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + 2t(\frac{xx_0}{a^2} + \frac{yy_0}{b^2}) + t^2(\frac{x^2}{a^2} + \frac{y^2}{b^2}) = (1 + t)^2,$$

or

$$(\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2} - 1) + 2t(\frac{xx_0}{a^2} + \frac{yy_0}{b^2} - 1) + t^2(\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1) = 0.$$

The point (x, y) will be on the tangent line to the ellipse if the roots of the equation for t are multiple, i.e. the discriminant of the equation is equal to zero. Hence, to get the equation of the tangent lines it is necessary to equate to zero the discriminant of the equation for t:

$$(\frac{x_0^2}{a^2} + \frac{y_0^2}{b^2} - 1)(\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1) - (\frac{xx_0}{a^2} + \frac{yy_0}{b^2} - 1)^2 = 0.$$

The equations of tangent lines to a hyperbola and parabola have an analogous form. Let us note here that the straight line

$$\frac{xx_0}{a^2} + \frac{yy_0}{b^2} = 1$$

passes through the points of tangency.

Thus far, we have given separate descriptions for the major distinguishing curved boundary meshes. Here we describe how these analysis are combined to form a boundary check procedure by an arbitrary set of boundary meshes. The check process is co-ordinated by the procedure shown in Figure (4.4).

4.4 Example: Simple Channel Flow

For the non-coordinate aligned channel flow, it is easy to determine the location of the centre of a mesh which lies in the channel or not. If the centre of the cell is located the same side of the both wall boundaries, or the location is located on one side and also located on the other wall boundary, this mesh will be a cut-cell and its half volume will locate outside the boundaries. We will cut out one quadrant in the next refinement.

As discussed above, redundant grid cells must be discarded during refinement of the grid. The problem is how to determine if the cell is in the calculation domain or out of it. To solve this problem, an algorithm to distinguish the position of the meshes near the boundaries is used. At this stage, we only need to consider the simple case: the linear boundary system. For example, the non-coordinate axes aligned with the steady state channel flow has two straight line boundaries (Figure 4.5).

The equation of the lower boundary is:

$$\eta = \frac{1}{2}\xi \tag{4.11}$$

The equation of the upper boundary is:

$$\eta = \frac{1}{2}\xi + \frac{b}{cos\alpha} \tag{4.12}$$

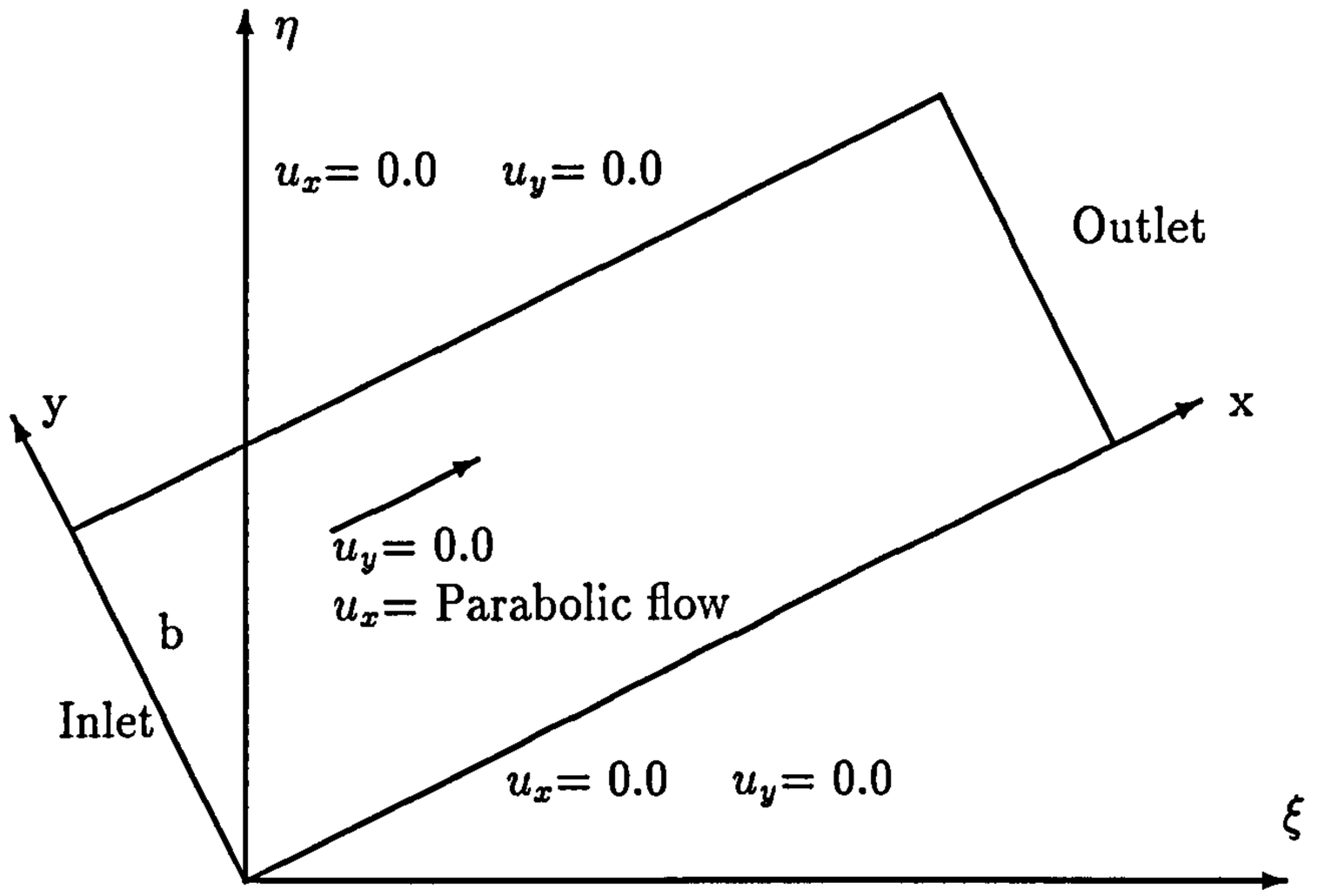


Figure 4.5: Picture to illustrate the channel flow.

We introduce the above algorithm to distinguish the positions of the grid near the boundary areas in the PAMG code. A pseudo-code procedure is given in Figure (4.6). There are two kinds of situations need to be considered:

A. If the tangent of the angle α formed by the straight line with the x-axis is positive, the centre point of all meshes should be checked. For the upper boundary, if the centre point of a cell is located above the two straight boundary lines, the mesh is a cut-cell and the second quadrant should be thrown away. If the centre point of a grid is located below the two boundary lines, the mesh should also be a cut-cell and the fourth quadrant should be discarded.

B. If the tangent of the angle α formed by the straight line with the x-axis is negative, the centre point of all meshes should be checked. If the centre point of a mesh is located above of the two straight line, the mesh should be a cut cell, and the first quadrant should be cut out. If the

Procedure $Bdry\_flag(x_j, y_j, Value_T, Value_B, \alpha)$
If $(Value_T.AND.Value_B) \geq 0.0$ then
If $(\alpha \leq 90^0)$ $bdry\_flag$(id,3) = 3
If $(\alpha \geq 90^0)$ $bdry\_flag$(id,4) = 4
End if
If $(Value_T.AND.Value_B) \leq 0.0$ then
If $(\alpha \leq 90^0)$ $bdry\_flag$(id,2) = 2
If $(\alpha \geq 90^0)$ $bdry\_flag$(id,1) = 1
End if
Enf procedure

Figure 4.6: Pseudo-code procedure to set up cut-cell flags for the boundary meshes.

centre point of a cell is below the two straight boundary line, the mesh should be a cut-cell and its third quadrant should be cut out.

4.5 Error Estimate of The Cut Cell

In regions of smooth flow, the criterion used for refining the grid is an estimate of the error in the solution on the finest existing grid in that region. Although there is no theory for equations of mixed type, in the purely elliptic or purely hyperbolic case there are estimates for the global error in the solution in terms of the local truncation error. Accordingly, the local truncation error in the solution will be estimated using ideas similar to Richardson extrapolation or the deferred correction method [13].

After the Cartesian cut-cell has been introduced, one question is created in this new procedure. How does one estimate the errors when the Cartesian cut-cell is used to treat the arbitrarily complex boundary conditions. The accuracy of non-uniform grids in one dimensional is discused in appendix B. To solve this problem, a suitable numerical estimate method must be found.

The differentials of the dependent variables appearing in partial differential equations must be expressed as approximate expressions, so that a powerful computer can be employed to obtain a solution. There are two methods for approximating the differentials of a function $f(x, y)$ in numerical analysis. One approximation often used is the Taylor series expansion of the function. Another method is the use of a polynomial of degree n. The second method is easily used to treatment the unequal step-size problems.

4.5.1 Finite Difference by Polynomials

The coefficients of the polynomial are computed by substitution of data from a series of usually equally spaced points of the independent variable. The approximate values of the derivatives are computed from the polynomial. For example, consider a second-order polynomial,

$$f(x) = Ax^2 + Bx + C \tag{4.13}$$

Select the origin at x_i. Thus, $x_i = 0$, $x_{i+1} = \Delta x$, and $x_{i+2} = 2\Delta x$ and the values of the function f at these locations are, f(x_i) = f_i, $f(x_{i+1}) = f_{i+1}$, and f(x_{i+2})= f_{i+2}. Thus,

$$f_i = Ax_i^2 + Bx_i + C = C$$

$$f_{i+1} = Ax_{i+1}^2 + Bx_{i+1} + C = A(\Delta x)^2 + B(\Delta x) + C$$

$$f_{i+2} = Ax_{i+2}^2 + Bx_{i+2} + C = A(2\Delta x)^2 + B(2\Delta x) + C$$

From which it follows that

$$C = f_i$$

$$B = \frac{-f_{i+2} + 4f_{i+1} - 3f_i}{2(\Delta x)}$$

and

$$A = \frac{f_{i+2} - 2f_{i+1} + f_i}{2(\Delta x)^2}$$

Now computing the first derivative of f, yields:

$$\frac{\partial f}{\partial x} = 2Ax + B,$$

and at $x_i = 0$, we have

$$\frac{\partial f}{\partial x} = B.$$

Therefore

$$\frac{\partial f}{\partial x} = \frac{-f_{i+2} + 4f_{i+1} - 3f_i}{2(\Delta x)}, \tag{4.14}$$

which is identical to the second-order accurate forward difference expression. Note that this approximation is classified as second-order accurate for $\frac{\partial f}{\partial x}$ in the accuracy analysis of the Taylor series expansion. If the spacing of the points $i, i+1, i+2$ is not identical, a finite difference approximation of the derivative is found by the same procedure. Assume $x_i = 0$, $x_{i+1} = \Delta x$, $x_{i+2} = (1+\alpha)\Delta x$, then,

$$f_i = Ax_i^2 + Bx_i + C = C$$

$$f_{i+1} = Ax_{i+1}^2 + Bx_{i+1} + C = A(\Delta x)^2 + B(\Delta x) + C$$

$$f_{i+2} = A(1+\alpha)^2 x_{i+2}^2 + B(1+\alpha)x_{i+2} + C = A(2\Delta x)^2 + B(2\Delta x) + C$$

Consequently,

$$C = f_i$$

$$B = \frac{-f_{i+2} + (1+\alpha)^2 f_{i+1} - (\alpha^2 + 2\alpha) f_i}{\alpha(1+\alpha)\Delta x}$$

and

$$A = \frac{f_{i+2} - (1+\alpha) f_{i+1} + \alpha f_i}{\alpha(1+\alpha)(\Delta x)^2}$$

therefore,

$$\left.\frac{\partial f}{\partial x}\right|_i = \frac{-f_{i+2} + (1+\alpha)^2 f_{i+1} - (\alpha^2 + 2\alpha) f_i}{\alpha(1+\alpha)\Delta x} \tag{4.15}$$

To investigate the truncation error of the above equation (4.15), the Taylor series expansion has been used. Since $x_i = 0$, $x_{i+1} = \Delta x$, $x_{i+2} = (1+\alpha)\Delta x$, the Taylor series expansions of $f(x + \Delta x)$ and $f(x + (1+\alpha)\Delta x)$ about x are given by:

$$f_{i+1} = f(x + \Delta x)$$

$$= f(x) + (\Delta x)\frac{\partial f}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \ldots,$$

$$f_{i+2} = f(x + (1+\alpha)\Delta x)$$

$$= f(x) + (1+\alpha)(\Delta x)\frac{\partial f}{\partial x} + \frac{((1+\alpha)\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} + \frac{((1+\alpha)\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \ldots$$

In order to get the truncation error of equation (4.15), we should rearrange f_i, f_{i+1}, and f_{i+2} with the coefficient in equation (4.15), i. e.:

$$-\alpha(\alpha+2)f_i = -\alpha(\alpha+2)f(x)$$

$$(1+\alpha)^2 f_{i+1} = (1+\alpha)^2 f(x+\Delta x)$$

$$= (1+\alpha)^2 f(x) + (1+\alpha)^2(\Delta x)\frac{\partial f}{\partial x} + (1+\alpha)^2\frac{(\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} + (1+\alpha)^2\frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \ldots$$

$$-f_{i+2} = -f(x + (1+\alpha)\Delta x)$$

$$= -f(x) - (1+\alpha)(\Delta x)\frac{\partial f}{\partial x} - \frac{[(1+\alpha)\Delta x]^2}{2!}\frac{\partial^2 f}{\partial x^2} - \frac{[(1+\alpha)\Delta x]^3}{3!}\frac{\partial^3 f}{\partial x^3} + \ldots$$

Adding these three equations together, yields:,

$$-f_{i+2} + (1+\alpha)^2 f_{i+1} - \alpha(\alpha+2)f_i$$

$$= \alpha(\alpha+1)\Delta x\frac{\partial f}{\partial x} - \alpha(1+\alpha)^2\frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3}$$

rearranging this equation, we obtain:

$$\alpha(\alpha+1)\Delta x\frac{\partial f}{\partial x}$$

$$= -f_{i+2} + (1+\alpha)^2 f_{i+1} - \alpha(\alpha+2)f_i + \alpha(1+\alpha)^2\frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3}$$

So,

$$\left.\frac{\partial f}{\partial x}\right|_i = \frac{-f_{i+2} + (1+\alpha)^2 f_{i+1} - \alpha(\alpha+2)f_i}{\alpha(\alpha+1)\Delta x} + (1+\alpha)\frac{(\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3}$$

$$= \frac{-f_{i+2} + (1+\alpha)^2 f_{i+1} - \alpha(\alpha+2)f_i}{\alpha(\alpha+1)\Delta x} + O(\Delta x)^2 \qquad (4.16)$$

which is a second-order accurate approximation for the unequal stepsize expressions.

4.5.2 Error Estimate Near the Solid Wall

The wall is the most common boundary encountered in confined fluid flow problems. For laminar flows, the boundary layer exists as a result of the action of viscous shear τ within the fluid, and the shear stress is proportional to the velocity gradient, i. e.:

$$\tau = \mu \frac{du}{dy}$$

The wall force is entered into the discretized u-momentum equation as a source term. The wall shear stress value is obtained from:

$$\tau_w = \mu \frac{u_p}{\Delta y_p}$$

where u_p is the velocity at the grid node. Figure 4.7 illustrate that this formula is based on the assumption that the velocity is a linear function of distance from the wall in the case of laminar flow.

The shear force F_s is now given by

$$F_s = -\tau A_{cell}$$

$$= -\mu \frac{u_p}{\Delta y_p} A_{cell}$$

$$= -\frac{\mu}{\Delta y_p} A_{cell} u_p$$

where A_{cell} is the wall area of the control volume. so,

$$A_{cell} = A_s$$

$$\Delta y_p = \frac{1}{2} \Delta y$$

The shear force F_s is then given by:

$$F_s = -\frac{2\mu A_s}{\Delta y} u_p$$

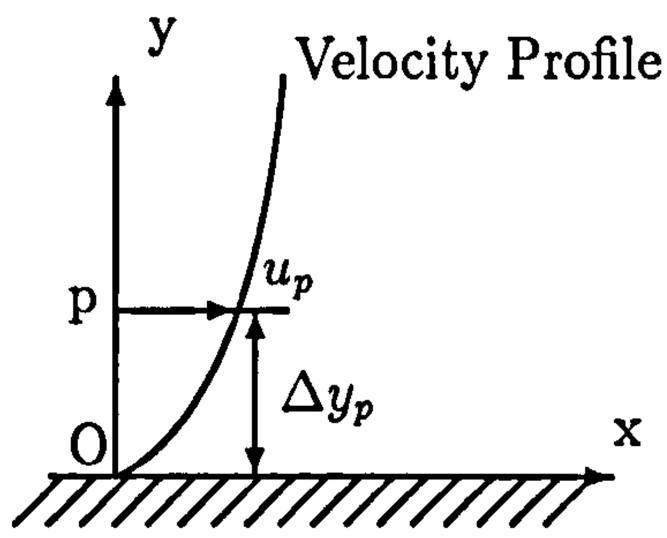


Figure 4.7: Velocity distribution at a wall.

Error estimate with solid walls

To deal with the accuracy issue near solid walls, a second-order polynomial equation has been considered, as:

$$u(y) = Ay^2 + By + C$$

which is shown in figure 4.6. Select the origin at y_i,

$$y_i = 0, \qquad y_{i+1} = \frac{1}{2}\Delta y, \qquad y_{i+2} = \frac{3}{2}\Delta y$$

the values of the function u at these locations are:

$$u(y_i) = u_i,$$

$$u(y_{i+1}) = u_{i+1},$$

$$u(y_{i+2}) = u_{i+2}.$$

Thus

$$u_i = Ay_i^2 + By_i + C = C,$$

$$u_{i+1} = \frac{1}{4}A\Delta y^2 + \frac{1}{2}B\Delta y + C,$$

$$u_{i+2} = \frac{9}{4}A\Delta y^2 + \frac{3}{2}B\Delta y + C,$$

so, we have

$$C = u_i,$$

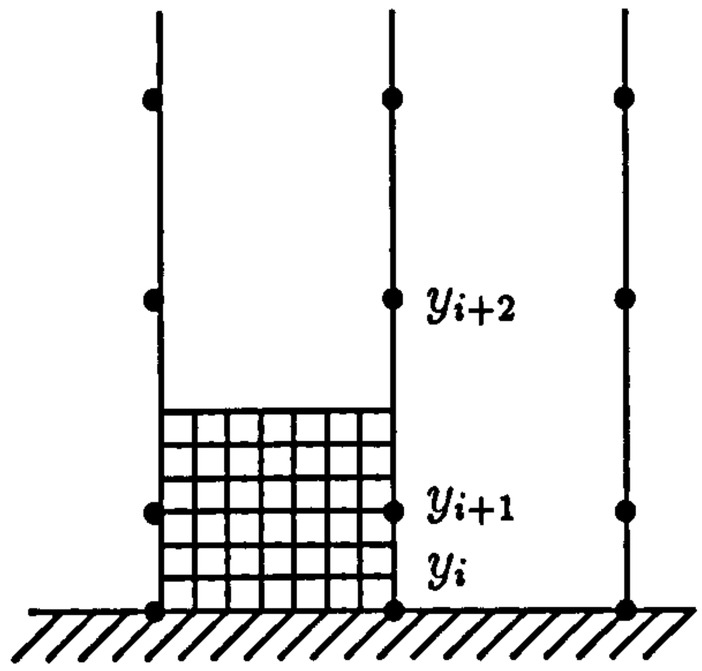


Figure 4.8: Velocity cell at a wall boundary.

$$B = \frac{9u_{i+1} - u_{i+2} - 8u_i}{3\Delta y},$$

$$A = \frac{2ui + 2 - 6u_{i+1} + 4u_i}{3(\Delta y)^2},$$

Now computing the first derivative of $u(y)$, we obtain:

$$\left.\frac{\partial u}{\partial y}\right|_i = 2Ay + B,$$

and at $y_i = 0$,

$$\left.\frac{\partial u}{\partial y}\right|_i = B.$$

Hence,

$$\frac{\partial u}{\partial y} = \frac{9u_{i+1} - u_{i+2} - 8u_i}{3\Delta y}$$

which is identical to the second-order accurate forward difference expression. If we use the three point values to determine the velocity gradient at the wall boundaries for the unfitted grid, it has the second-order accuracy.

4.5.3 Richardson Extrapolation

The method, suggested by Richardson, know as Richardson Extrapolation, is extremely useful when there is a reliable estimate of the discretization error as a function

of the mesh length. It is a method for obtaining additional accuracy from a sequence of calculations.

Let U be the exact solution of a differential equation and u_1, u_2 the approximate solutions at the same mesh-point for mesh lengths h and H respectively. we have:

$$U = u_1 + h^2 g + O(h^3) \tag{4.17}$$

$$U = u_2 + H^2 g + O(H^3) \tag{4.18}$$

Then simple algebra gives a new approximation:

$$U = \frac{H^2 u_1 - h^2 u_2}{H^2 - h^2} + O(h^3) \tag{4.19}$$

If we have $H = 2h$, then this can be written as

$$U = u_1 + \frac{u_1 - u_2}{3} + O(h^3). \tag{4.20}$$

For the Navier-Stokes equations, the discretization error for a rectangular region with smooth boundary-values is proportional to h^2. An estimate can be found from three approximate solutions at the same grid-point for the value of proportionality to h^p. Taking

$$h_1 = \frac{1}{2} h_2 = \frac{1}{4} h_4 \tag{4.21}$$

gives

$$\frac{u_2 - u_1}{u_3 - u_2} = 2^p \tag{4.22}$$

For the Navier-Stokes equation, $h^p = h^2$, so the error is reduced by a factor of four. Due to the use of the Cartesian cut-cell, additional errors maybe created in this procedure, so the error reduction may be less than 4.

Chapter 5

The CC-PAMG Algorithm

In chapter 4, we introduced a new algorithm (Cartesian cut-cell method) to solve the Navier-Stokes equations in complicated geometries. This requires solid wall boundary conditions in the irregular grid cells near the boundary. Since these cut cells may be orders of magnitude smaller than the regular grids. The accuracy is primary concern. The methods to determine the cut-cell are given and the error estimate of the cut-cell are investigated with three different methods. In this chapter, we will give more details of the boundary treatment which includes the simple and complex boundary conditions. The improve of the grid transfer operators is also be demonstrated.

5.1 The CC-PAMG Algorithm - Modification of The Grid Transfer Operators

In this section, we conclude our analysis of the different parts of the multigrid algorithm implemented in CC-PAMG, by investigating the efficiency of the inter-grid transfers. The results are based on second-order interpolation of the pressure and third-order accuracy of momentum equations corrections during the prolongation stage of the FAS computations, while the original PAMG results presented are based on first order interpolation of the pressure and first order interpolation of the momentum equations corrections near the walls.

Although transfer operators in a multigrid method do not influence the solution on the fine grid, they may have a very significant bearing on the convergence factors of the method since interpolation errors will need to be eliminated during relaxation. It is therefore legitimate to ask whether the absence of grid independent convergence is due to inaccurate prolongation of the cell centred quantities which is performed to a low order of accuracy. The prolongation of the velocities, being located on edges, is accurate up to second order as explained in Section 5.1.2 and hence should not cause any problems on fine grids.

5.1.1 Restriction of the Fine Grid Approximation

We have to restrict three types of quantities: the vertical and horizontal velocities, which are defined at cell edges and the pressures, which are defined at cell centres. For the velocities, second order linear interpolation is used. Letting $if = 2ic$ and $jf = 2jc$, the coarse grid values are defined in terms of the fine grid values by the following relationships:

$$u_{ic+1/2,jc} = \frac{1}{2}(u_{if+1/2,jf-1} + u_{if+1/2,jf}), \tag{5.1}$$

$$u_{ic,jc+1/2} = \frac{1}{2}(v_{if-1,jf-1/2} + v_{if,jf+1/2}), \tag{5.2}$$

where the subscripts c and f refer to the coarse and fine grid values respectively. Similarly, bilinear interpolation is used for the pressure:

$$p_{ic,ic} = \frac{1}{4}(p_{if-1,jf-1} + p_{jf-1,jf} + p_{jf,jf-1} + p_{jf,jf}) \tag{5.3}$$

This last relationship applies for interior points only but it is the only restriction formula necessary for the pressures. First, no pressure boundary conditions are necessary since the continuity equation is satisfied directly using the velocities rather than a pressure based Poisson equation. At solid walls, velocities are fixed so that no transfer is necessary. Finally, for Neumann conditions, the formula for interior nodes are used and boundary conditions are applied just after the transfer.

For a Cartesian cut cell, the restriction formula necessary for the pressure is no longer applicable to this situation. The restriction formula for the pressures must be modified. In this simple case, only a quarter has been cut out, so the bilinear interpolation is changed for the pressure:

$$p_{ic,ic} = \frac{1}{3}(p_{if-1,jf-1} + p_{jf-1,jf} + p_{jf,jf-1}) \tag{5.4}$$

if the $p_{if,jf}$ has been cut.

The residuals of the momentum equations are defined on cell edges. It is therefore possible to use the same operators as for the restriction of the velocities. The horizontal momentum residuals are restricted using Equation 5.1 while the vertical momentum equations are restricted using Equation 5.2. The pressure operator (Equation 5.3) is used for the restriction of continuity residuals to the coarse grid, since they are computed at cell centres.

5.1.2 Prolongation of the Corrections

We need to interpolate both cell-centred and edge-defined coarse grid corrections. It is possible to transfer the velocities using standard (second order accurate) operators as follows:

$$u_{if+1/2,jf} = \frac{3}{4}u_{ic+1/2,jc} + \frac{1}{4}u_{ic+1/2,jc+1}, \tag{5.5}$$

$$u_{if+1/2,jf-1} = \frac{3}{4}u_{ic+1/2,jc} + \frac{1}{4}u_{ic+1/2,jc-1}, \tag{5.6}$$

which with $2if = ic$, $2jf = jc$. We note that these operators are not conservative: the numerical mass flux on the coarse grid is not equal to that on the fine grid. For computations involving only uniform grids, this is not a problem since the solution is only sought on the finest grid. Any mass variation during the grid transfers is incorporated in the defect.

For adaptive computations with composite grids involving several levels, it is necessary to conserve mass fluxes at grid interfaces, otherwise the discrete mass equation may not admit a solution. This requirement will be satisfied by designing conservative interpolation schemes for both the restrictions of velocity values and the prolongation of velocity corrections. We notice that the restriction operators described above conserve the mass fluxes exactly, and then developed a prolongation operator for velocity corrections which shares the same property. So that,

$$u_{if+1/2,jf} = u_{ic+1/2,jc} + \frac{1}{8}(u_{ic+1/2,jc+1} - u_{ic+1/2,jc-1}), \tag{5.7}$$

$$u_{if+1/2,jf-1} = u_{ic+1/2,jc} - \frac{1}{8}(u_{ic+1/2,jc+1} - u_{ic+1/2,jc-1}), \tag{5.8}$$

Noting that in PAMG, we need to interpolate the corrections rather than the solution values to the fine grid, and using the notation of the previous. The following operators for the prolongation of the velocities in original PAMG code have been applied:

$$\Delta u_{if+1/2,jf} = \Delta u_{ic+1/2,jc} + \frac{1}{8}(\Delta u_{ic+1/2,jc+1} - \Delta u_{ic+1/2,jc-1}) \tag{5.9}$$

$$\Delta u_{if+1/2,jf-1} = \Delta u_{ic+1/2,jc} - \frac{1}{8}(\Delta u_{ic+1/2,jc+1} - \Delta u_{ic+1/2,jc-1}) \tag{5.10}$$

$$\Delta u_{if-1/2,jf} = \frac{1}{2}(\Delta u_{ic+1/2,jc} + \Delta u_{ic-1/2,jc}) + \frac{1}{16}(\Delta u_{ic+1/2,jc+1} + \Delta u_{ic-1/2,jc+1} - \Delta u_{ic+1/2,jc-1} - \Delta u_{ic-1/2,jc-1}) \tag{5.11}$$

$$\Delta u_{if-1/2,jf-1} = \frac{1}{2}(\Delta u_{ic+1/2,jc} + \Delta u_{ic-1/2,jc}) - \frac{1}{16}(\Delta u_{ic+1/2,jc+1} + \Delta u_{ic-1/2,jc+1} - \Delta u_{ic+1/2,jc-1} - \Delta u_{ic-1/2,jc-1}) \tag{5.12}$$

The transfer operators for the vertical velocity are defined in a very similar way:

$$\Delta v_{if,jf+1/2} = \Delta v_{ic,jc+1/2} + \frac{1}{8}(\Delta v_{ic+1,jc+1/2} - \Delta v_{ic-1,jc+1/2}) \tag{5.13}$$

$$\Delta v_{if-1,jf+1/2} = \Delta v_{ic,jc+1/2} - \frac{1}{8}(\Delta v_{ic+1jc+1/2} + -\Delta v_{ic-1,jc+1/2}) \tag{5.14}$$

$$\Delta v_{if,jf-1/2} = \frac{1}{2}(\Delta v_{ic,jc-1/2} + \Delta v_{ic,jc+1/2}) + \frac{1}{16}(\Delta v_{ic+1,jc+1/2} +$$

$$\Delta v_{ic+1,jc-1/2} - \Delta v_{ic-1,jc+1/2} - \Delta v_{ic-1,jc-1/2}) \tag{5.15}$$

$$\Delta v_{if-1,jf-1/2} = \frac{1}{2}(\Delta v_{ic,jc+1/2} + \Delta v_{ic,jc-1/2}) - \frac{1}{16}(\Delta v_{ic+1,jc+1/2} +$$

$$\Delta v_{ic+1,jc-1/2} - \Delta v_{ic-1,jc+1/2} - \Delta v_{ic-1,jc-1/2}) \tag{5.16}$$

These formulas are modified near walls. For simplicity, the transfer operators for the horizontal and vertical velocity components are respectively defined as follows:

$$\Delta u_{if+1/2,jf} = \Delta u_{ic+1/2,jc} \tag{5.17}$$

or

$$\Delta u_{if+1/2,jf-1} = \Delta u_{ic+1/2,jc} \tag{5.18}$$

and

$$\Delta v_{if,jf+1/2} = \Delta v_{ic,jc+1/2} \tag{5.19}$$

or

$$\Delta v_{if-1,jf+1/2} = \Delta v_{ic,jc+1/2} \tag{5.20}$$

which depending on the top or bottom edges of each patches. Although the velocity corrections are first-order accurate at outside edges for each patch, it is overlapped by its adjacent patches which has a second-order accuracy for this points. However, if the patches are located near walls, the first-order accurate velocity corrections introduce errors into the prolongation procedure. To overcome this a new algorithm has been developed, and the Lagrange three-point interpolation algorithm has been introduced to deal with the velocity corrections near walls.

It is relatively straightforward to demonstrate the third-order accuracy of the Lagrange interpolation scheme for the convective flux at a mid-point cell face in a uniform grid of spacing Δy near walls (see Figure 5.1).

This scheme calculates the value u_{f1} at the finer grid point as:

$$u_{f_1} = \frac{5}{32}u_{C_5} + \frac{15}{16}u_{C_1} - \frac{3}{32}u_{C_3} \tag{5.21}$$

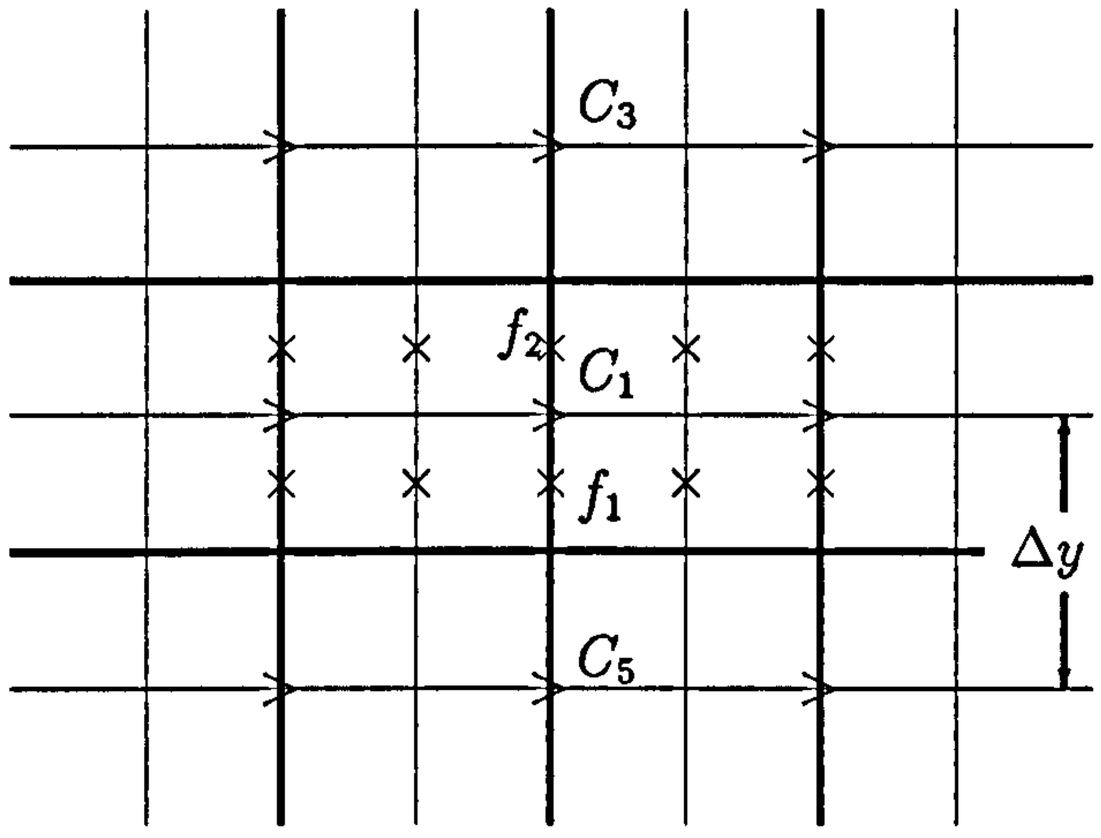


Figure 5.1: Prolongation at different grid levels.

Taylor series expansions about the finer grid node are given as:

$$u_{C_5} = u_{f_1} - \frac{3}{4}\Delta y(\frac{\partial u}{\partial y})_{f_1} + \frac{1}{2}(\frac{-3}{4}\Delta y)^2(\frac{\partial^2 u}{\partial y^2})_{f_1} + O(\Delta y^3) \tag{5.22}$$

$$u_{C_1} = u_{f_1} + \frac{1}{4}\Delta y(\frac{\partial u}{\partial y})_{f_1} + \frac{1}{2}(\frac{1}{4}\Delta y)^2(\frac{\partial^2 u}{\partial y^2})_{f_1} + O(\Delta y^3) \tag{5.23}$$

$$u_{C_3} = u_{f_1} + \frac{5}{4}\Delta y(\frac{\partial u}{\partial y})_{f_1} + \frac{1}{2}(\frac{5}{4}\Delta y)^2(\frac{\partial^2 u}{\partial y^2})_{f_1} + O(\Delta y^3) \tag{5.24}$$

by substituting equations (5.22) – (5.24) into equation (5.21), we obtain,

$$\frac{5}{32}u_{C_5} + \frac{15}{16}u_{C_1} - \frac{3}{32}u_{C_3} = u_{f_1} + O(\Delta y^3) \tag{5.25}$$

The terms involving orders Δy and Δy^2 cancel out in this uniform grid and this scheme is a third-order accurate approximation which is used near wall prolongation transfer operators in CC-PAMG.

It only remains to specify the prolongation operator for the pressure corrections. Since there are no difficulty to compute the definition of the operator, a simply choice

has been made in the original PAMG algorithm which are accurate to first order, namely:

$$\Delta p_{if,jf} = \Delta p_{ic,jc}, \tag{5.26}$$

$$\Delta p_{if,jf-1} = \Delta p_{ic,jc}, \tag{5.27}$$

$$\Delta p_{if-1,jf} = \Delta p_{ic,jc}, \tag{5.28}$$

$$\Delta p_{if-1,jf-1} = \Delta p_{ic,jc}. \tag{5.29}$$

Alternatively, the following second order formula can be adopted to improve the pressure accuracy in CC-PAMG algorithm:

$$\Delta p_{if,jf} = \frac{3}{4}\Delta p_{ic,jc} + \frac{1}{4}\Delta p_{ic+1,jc+1} \tag{5.30}$$

$$\Delta p_{if-1,jf} = \frac{3}{4}\Delta p_{ic,jc} + \frac{1}{4}\Delta p_{ic-1,jc+1} \tag{5.31}$$

$$\Delta p_{if,jf-1} = \frac{3}{4}\Delta p_{ic,jc} + \frac{1}{4}\Delta p_{ic+1,jc-1} \tag{5.32}$$

$$\Delta p_{if-1,jf-1} = \frac{3}{4}\Delta p_{ic,jc} + \frac{1}{4}\Delta p_{ic-1,jc-1} \tag{5.33}$$

5.2 Wall Boundary Condition Treatment

In general, a CFD problems are defined in terms of initial or boundary conditions. It is important that the user specifies these correctly and understands their role in the numerical algorithm. In transient problems the initial values of all the variables need to be specified at all solution points in the flow domain. Since this involves no special measures other than initialising the appropriate data arrays in the CFD code we do not need to discuss this topic further. The present section describes the implementation of the following most common boundary conditions in the discretized equation of the finite volume methods: wall boundary conditions.

The wall is the most common boundary encountered in confined flow problems. In this section we consider a solid wall parallel to the X-direction and parallel to the

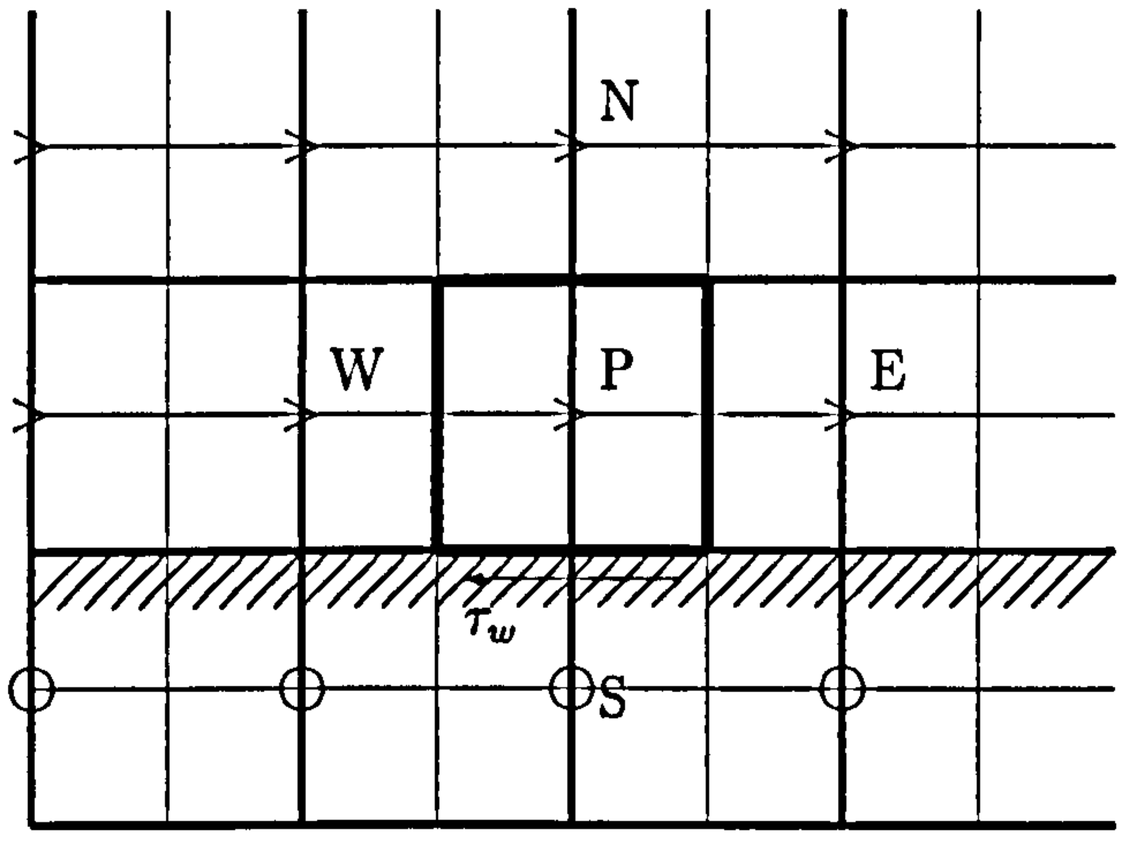


Figure 5.2: Horizontal Velocity Cell at a Bottom Wall Boundary. Where $'>'$ indicates the velocity along x direction and small circle means the velocity is zero.

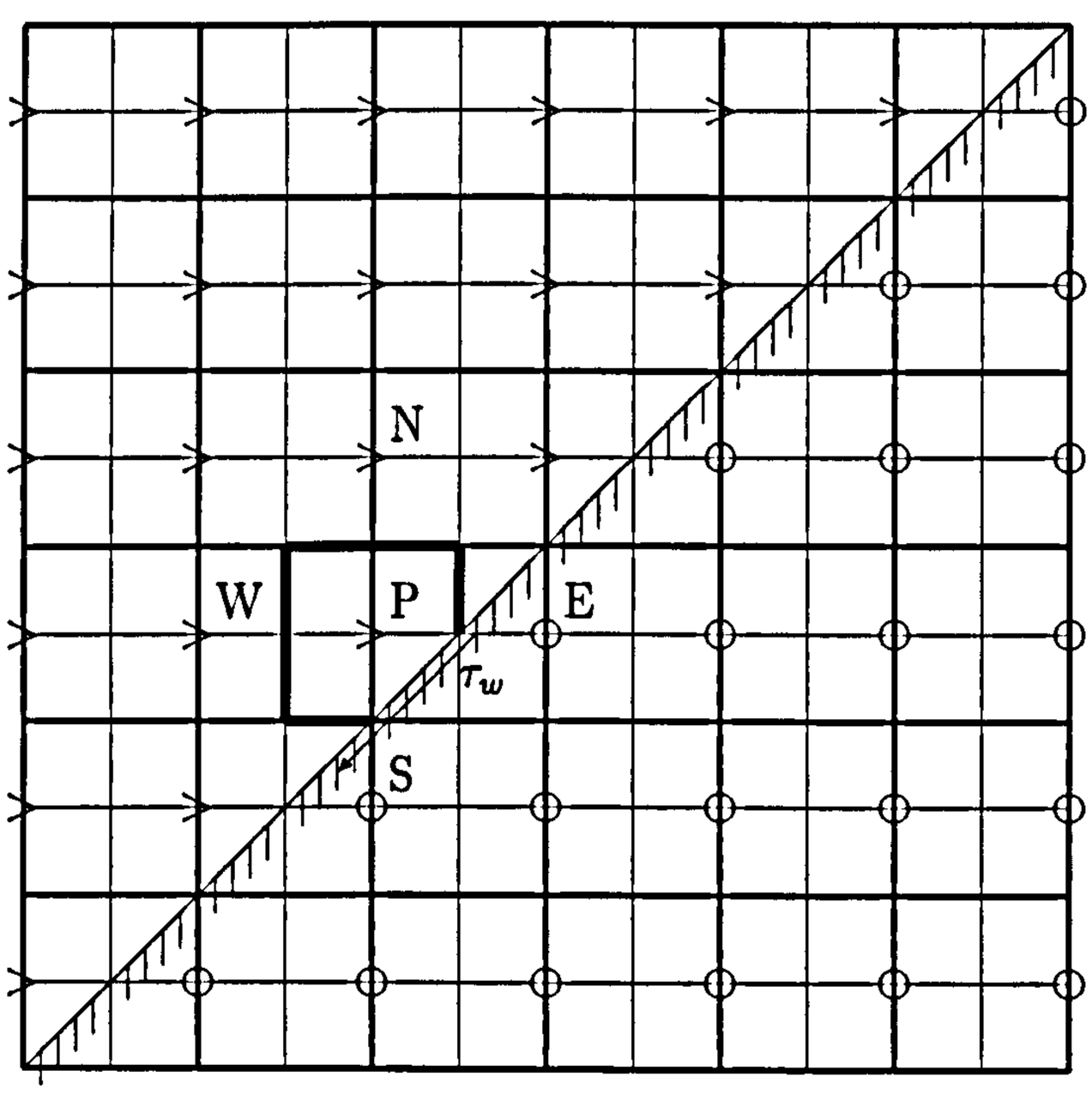


Figure 5.3: Horizontal Velocity Cell at a Non-aligned Aligned Bottom Wall Boundary

Y-direction. We also consider the non-aligned cases. Figure (5.2) to Figure (5.5) illustrate the grid details in the near wall regions for the u-velocity component and Figure (5.6) to Figure (5.9) illustrate the grid details for the v-velocity component (parallel to the wall and non-aligned aligned wall).

The non-slip condition ($u = 0.0, v = 0.0$) is the appropriate condition for the velocity components at solid walls. The normal component of the velocity can simply be set to zero at the boundary and the discretized momentum equation at the next v-cell in the flow can be evaluated without modification. There is no P-node (pressure) in the boundary, so it is also unnecessary to perform a pressure correction here.

In section 4.5.2, the error estimate near the solid wall has been made and the analysis solution has demonstrated that the wall boundary treatment is the second-order accurate forward difference expression. Consider the portion of the boundary shown from Figure (5.2) to Figure (5.9) and a typical boundary cell P, The formula for updating the value u_p is the two-dimensional analogue of Equation (3.23),

$$A_C^u u_{i+\frac{1}{2},j} = A_E^u u_{i+\frac{3}{2},j} + A_N^u u_{i+\frac{1}{2},j+1} + A_W^u u_{i-\frac{1}{2},j} + A_S^u u_{i+\frac{1}{2},j-1}$$

$$-\frac{1}{\Delta y}(p_{i+1,j} - p_{i,j})$$

Because there is a non-aligned wall in the grid P, the A_c^u is modified to:

$$A_C^u = A_E^u + A_N^u + A_W^u + A_S^u - S_p^u$$

as before:

$$A_E^u = max(|C_e^u|, D_e^u) - C_e^u,$$

$$A_W^u = max(|C_w^u|, D_w^u) + C_w^u,$$

$$A_N^u = max(|C_n^u|, D_n^U) - C_n^u,$$

$$A_S^u = max(|C_s^u|, D_s^u) + C_s^u,$$

Where S_p^u is the appropriate non-aligned source term in the u -equation defined by (see Figure 4.7):

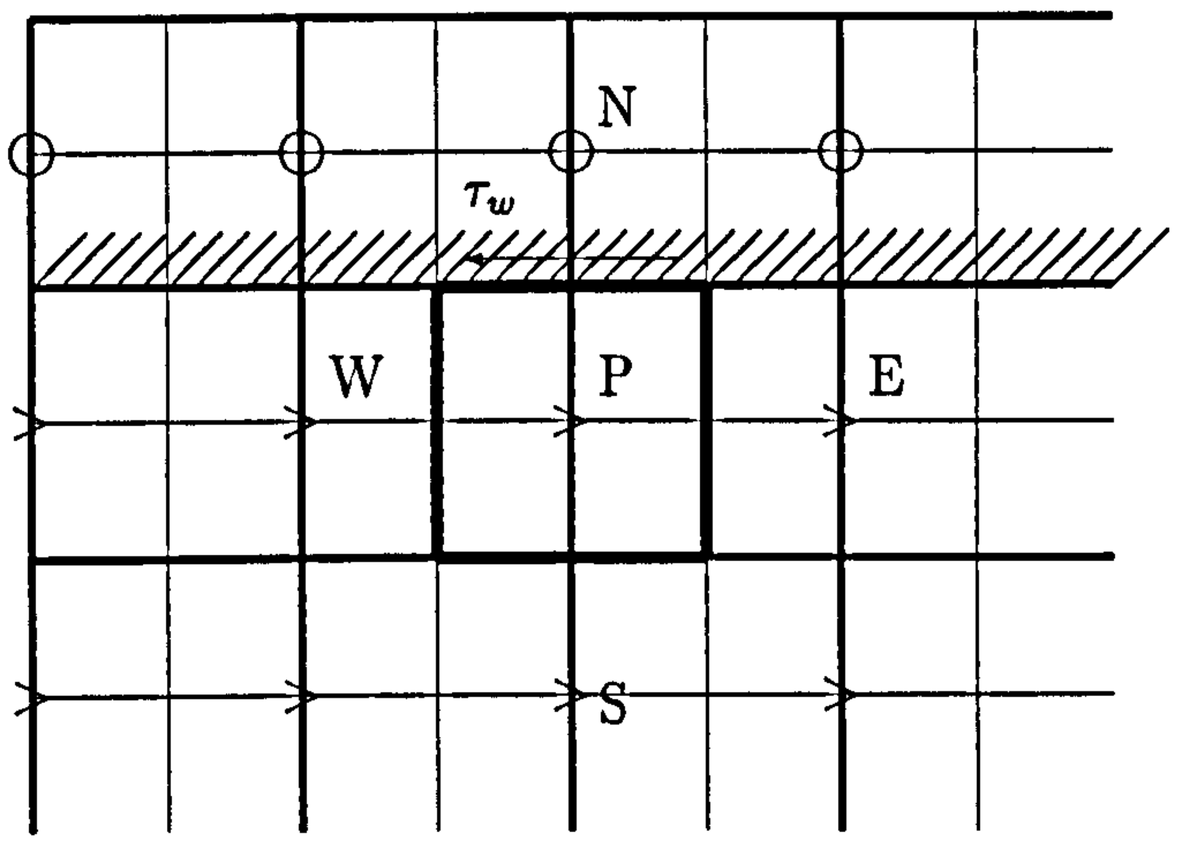


Figure 5.4: Horizontal Velocity Cell at a Upper Wall Boundary. Where $'>'$ indicates the velocity along x direction and small circle means the velocity is zero.

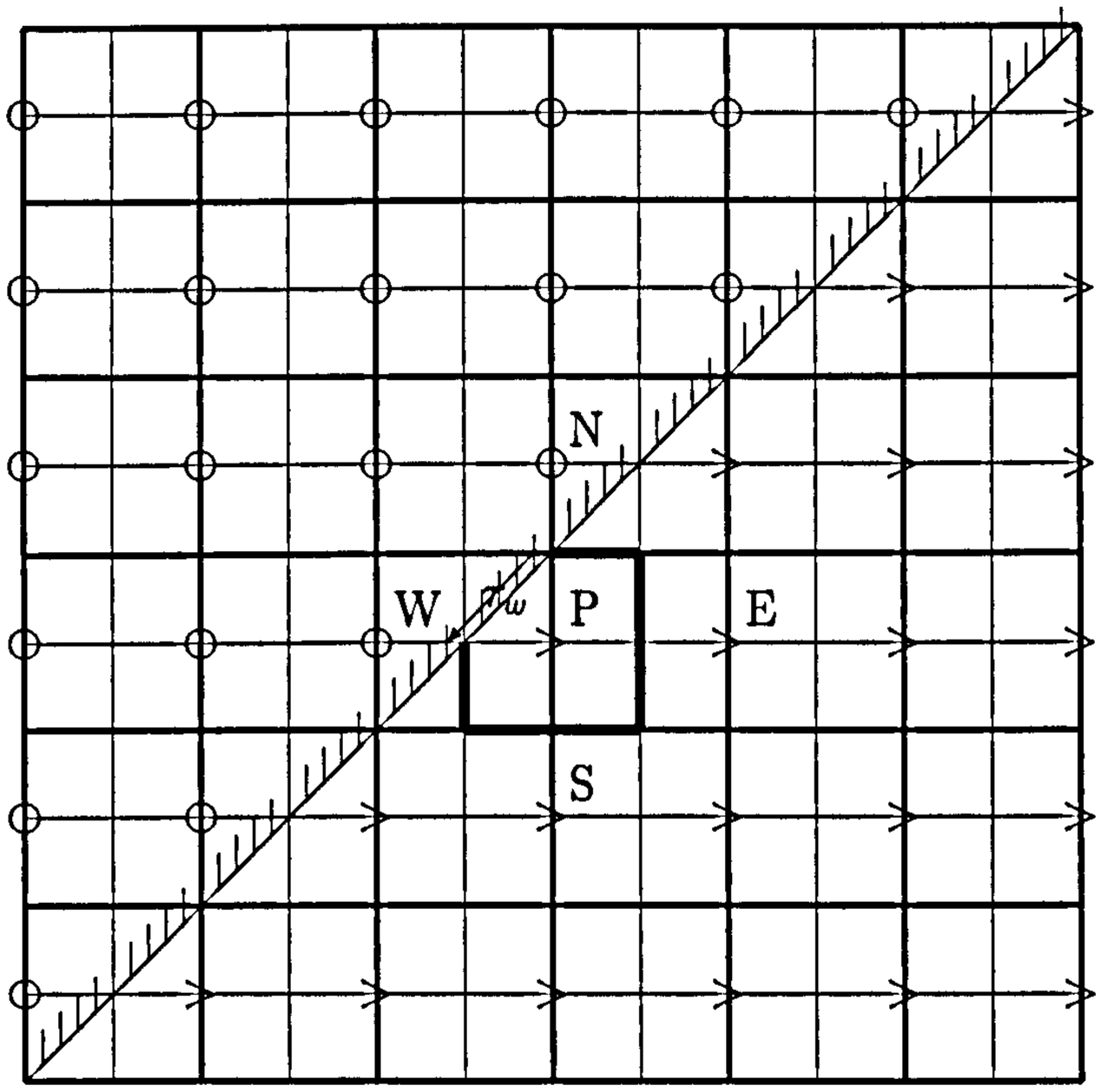


Figure 5.5: Horizontal Velocity Cell at a Non-aligned Upper Wall Boundary

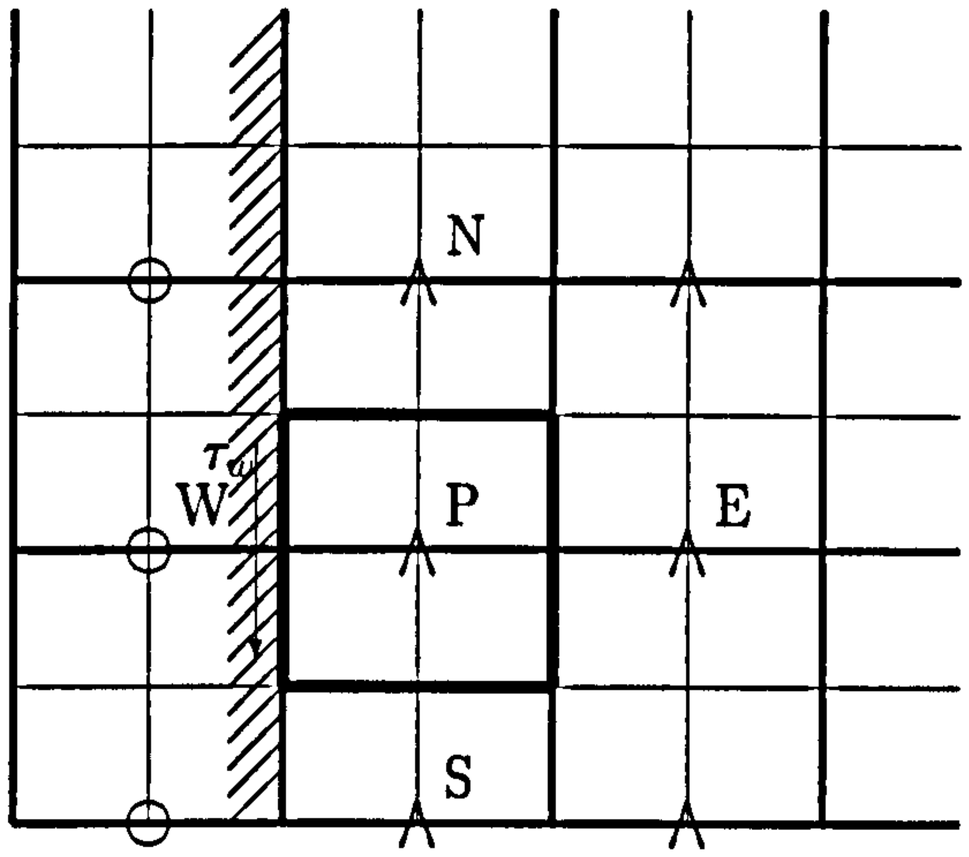


Figure 5.6: Vertical Velocity Cell at the Left Wall Boundary. Where '$\wedge$' indicates the velocity along y direction and small circle means the velocity is zero.

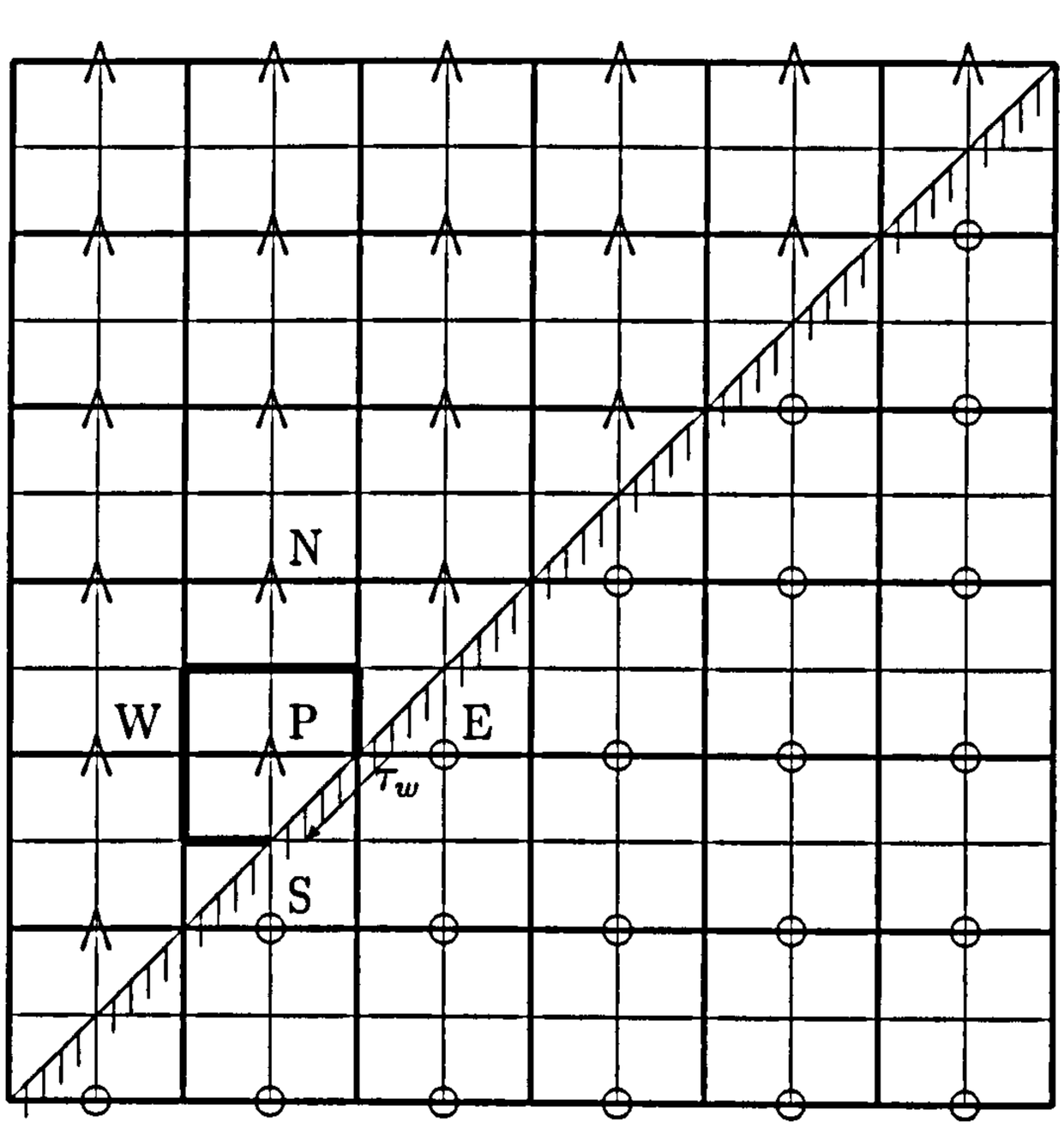


Figure 5.7: Vertical Velocity Cell at a Non-aligned Bottom Wall Boundary.

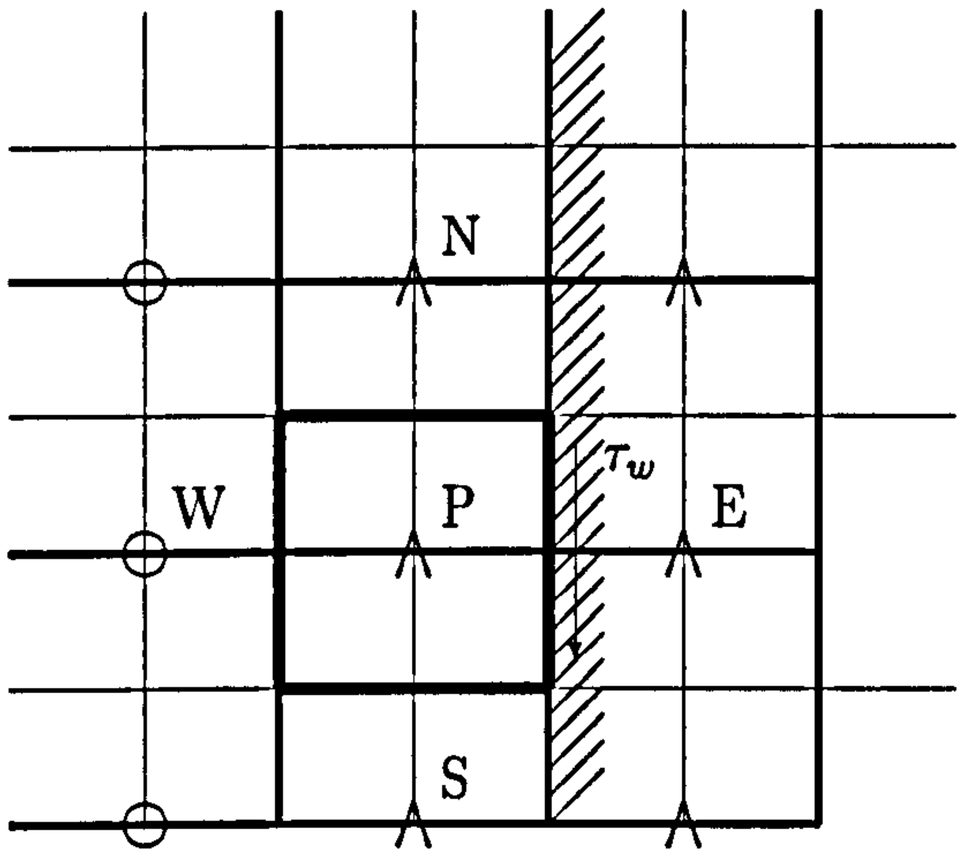


Figure 5.8: Vertical Velocity Cell at The Right Wall Boundary. Where $'\Lambda'$ indicates the velocity along y direction and small circle means the velocity is zero.

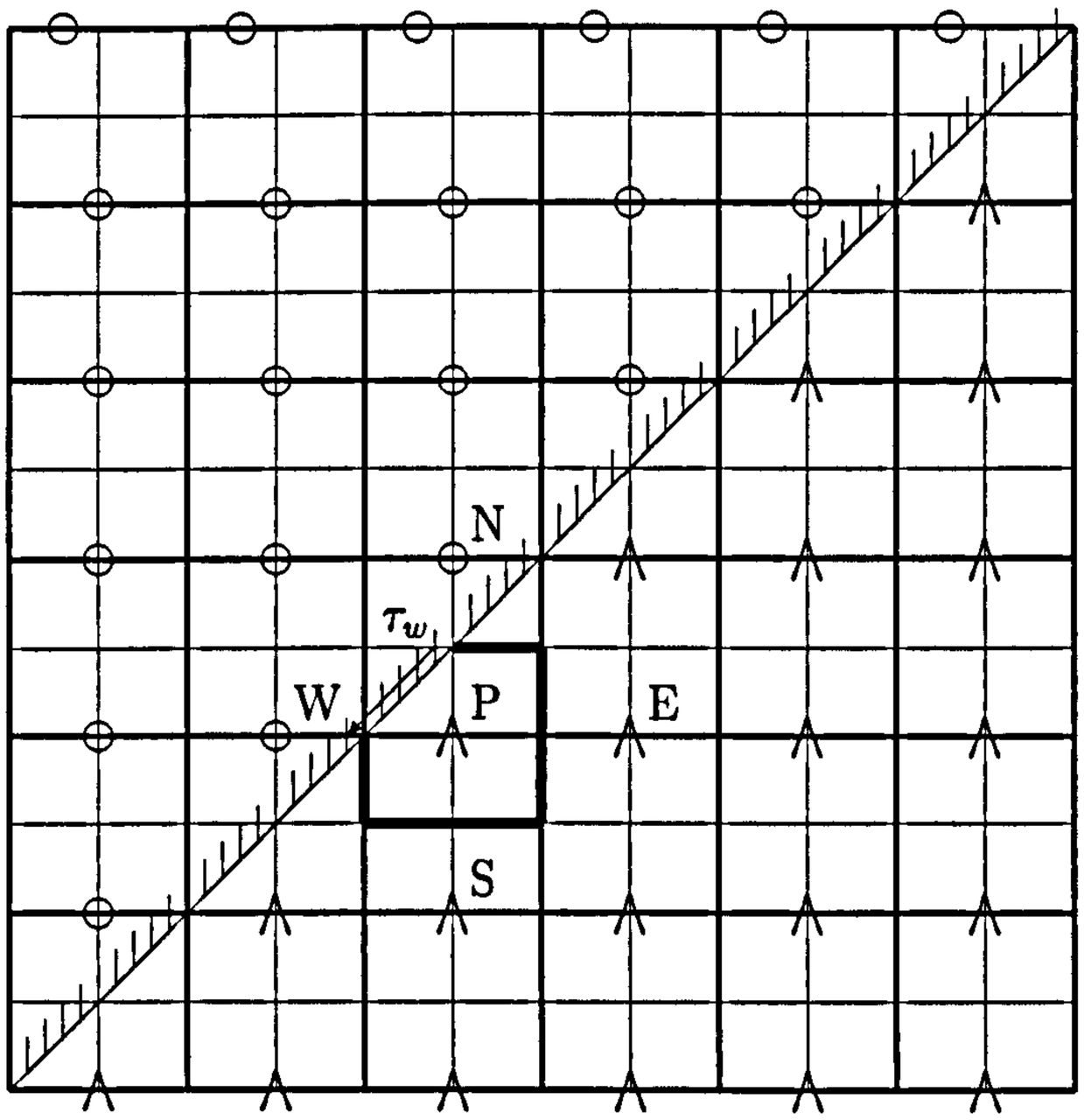


Figure 5.9: Vertical Velocity Cell at a Non-aligned Upper Wall Boundary

$$S_p = -\frac{\mu}{\Delta y_p} A_{cell}.$$

The shear force, F_u, for the non-coordinate aligned wall in the u-equation is:

$$F_u = -\mu \frac{u_p \cos\alpha}{\Delta x \sin\alpha} \sqrt{\Delta x^2 + \Delta y^2},$$

Where $A_{cell} = \sqrt{\frac{\Delta x^2}{4} + \frac{\Delta y^2}{4}}$ is the wall area of the control volume, and $\Delta y_p = \frac{\Delta x}{2} sin\alpha$ is the normal distance from the wall to the u -velocity at the grid node.

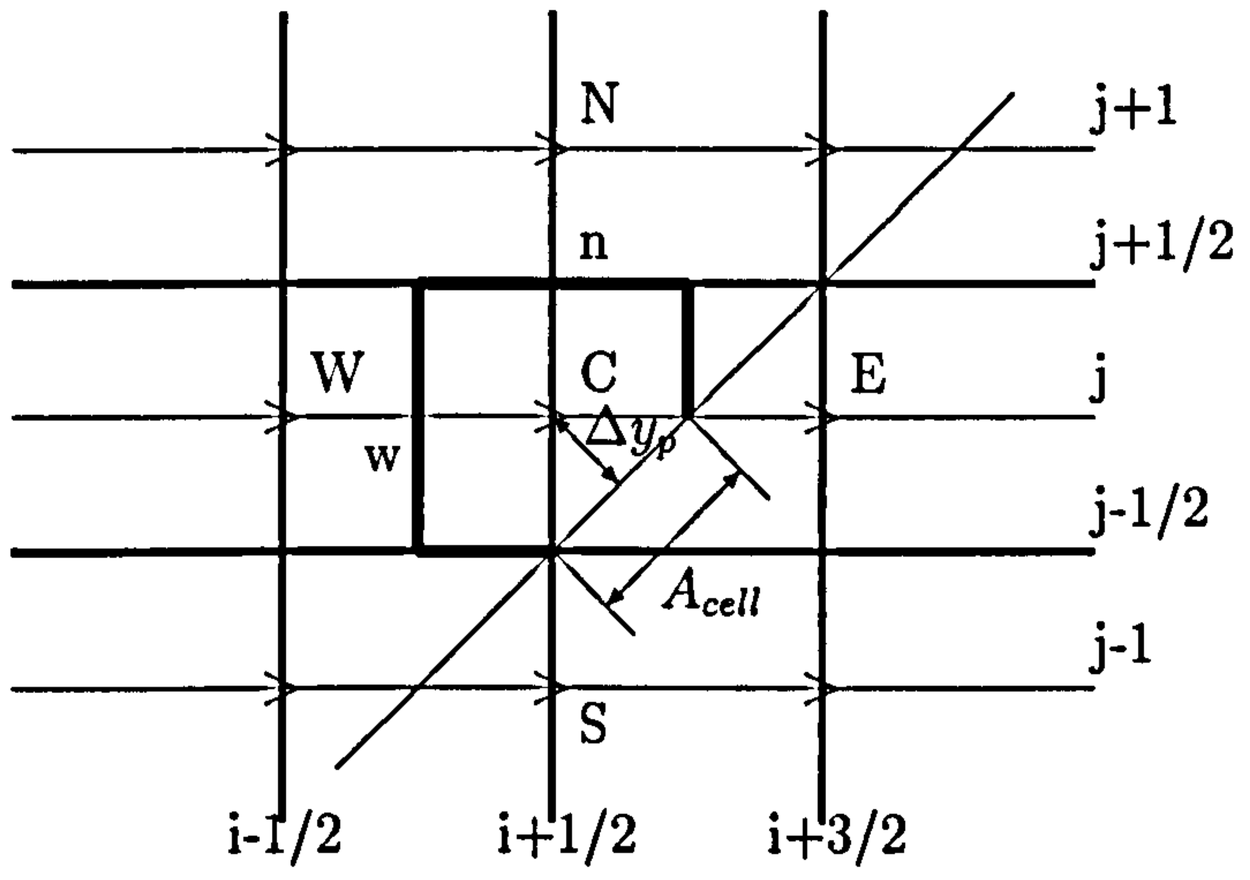


Figure 5.10: Cut-cell Control volume for the integration of the Navier-Stokes horizontal momentum equation

With the formula for updating the value in the v -equation, the appropriate non-aligned source term is defined by

$$S_p = -\frac{\mu}{\Delta x_p} A_{cell}$$

So the shear force F_v for the non-aligned wall in the v-equation is:

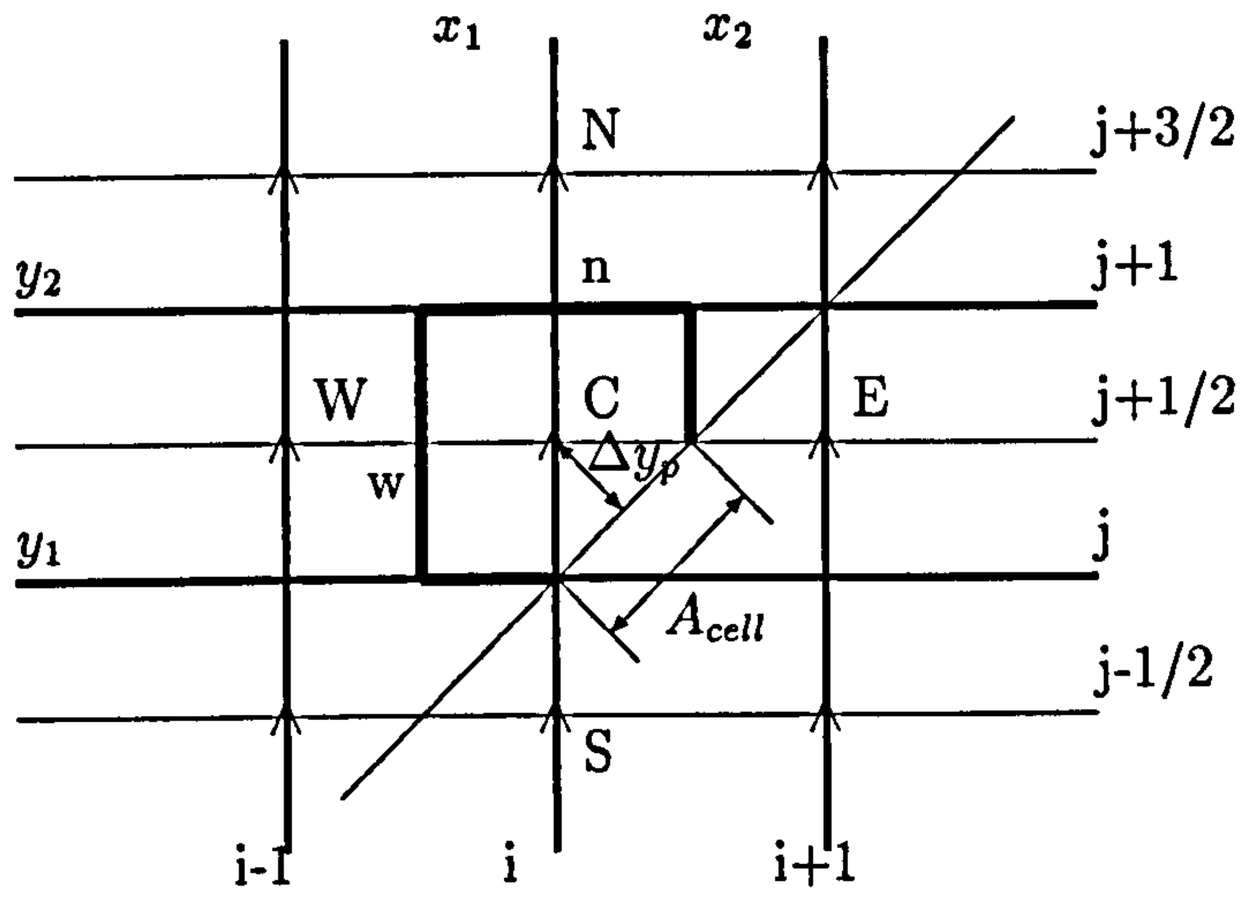


Figure 5.11: Cut-cell Control volume for the integration of the Navier-Stokes vertical momentum equation

$$F_v = -\mu \frac{v_p \sin\alpha}{\Delta x \sin\alpha} \sqrt{\Delta x^2 + \Delta y^2}$$

$$= -\mu \frac{v_p}{\Delta x} \sqrt{\Delta x^2 + \Delta y^2}$$

Where $A_{cell} = \sqrt{\frac{\Delta x^2}{4} + \frac{\Delta y^2}{4}}$ is the wall area of the control volume, and the $\Delta x_p = \frac{\Delta x}{2} sin\alpha$ is the normal distance from the wall to the v-velocity at the grid node. The angle α is determined by $tan^{-1}\left)\frac{\Delta y}{\Delta x}\right)$.

5.3 Numerical Results Comparison

We show one representation test case, a backward-facing step flows (Section 6.3). We also show the comparison results of the two algorithms at the same situation. The geometry and boundary condition we used are the same for this two different solvers. The initial conditions are the parabolic laminar flow in the inlet positions. The more

detail results can be found in Section 5.4.

Figure 5.12 and 5.13 show the comparisons of a backward-facing step flows with Reynolds number 100. We compared the average convergence ratio and the work units at different grid levels, which have error tolerance 10^{-6}. At the start level (grid 2), these two algorithms have nearly the same convergence rate and iteration cycle numbers. The convergence acceleration in PAMG is quickly increasing at beginning till level 4, then it is slightly decreasing and finally reach the same convergence rate as in grid 2. Contrary to the PAMG results, the convergence acceleration in CC-PAMG is fast and fast till the highest level (grid 7). In level 7, it needs 13 work units to reach the error tolerance in PAMG but in CC-PAMG it only needs 7 work units to reach the error tolerance. The CC-PAMG is more efficient than the original PAMG.

Figure 5.14 and 5.15 show the comparison of the backward-facing step flows with Reynolds number 500. They have the nearly the same results of comparison with the Reynolds number at 100.

This method can be extended to the more complicated computations with complex geometry. The improved transformation method and boundary conditions described here can also be used in the following chapters to treatment the complex geometry problems.

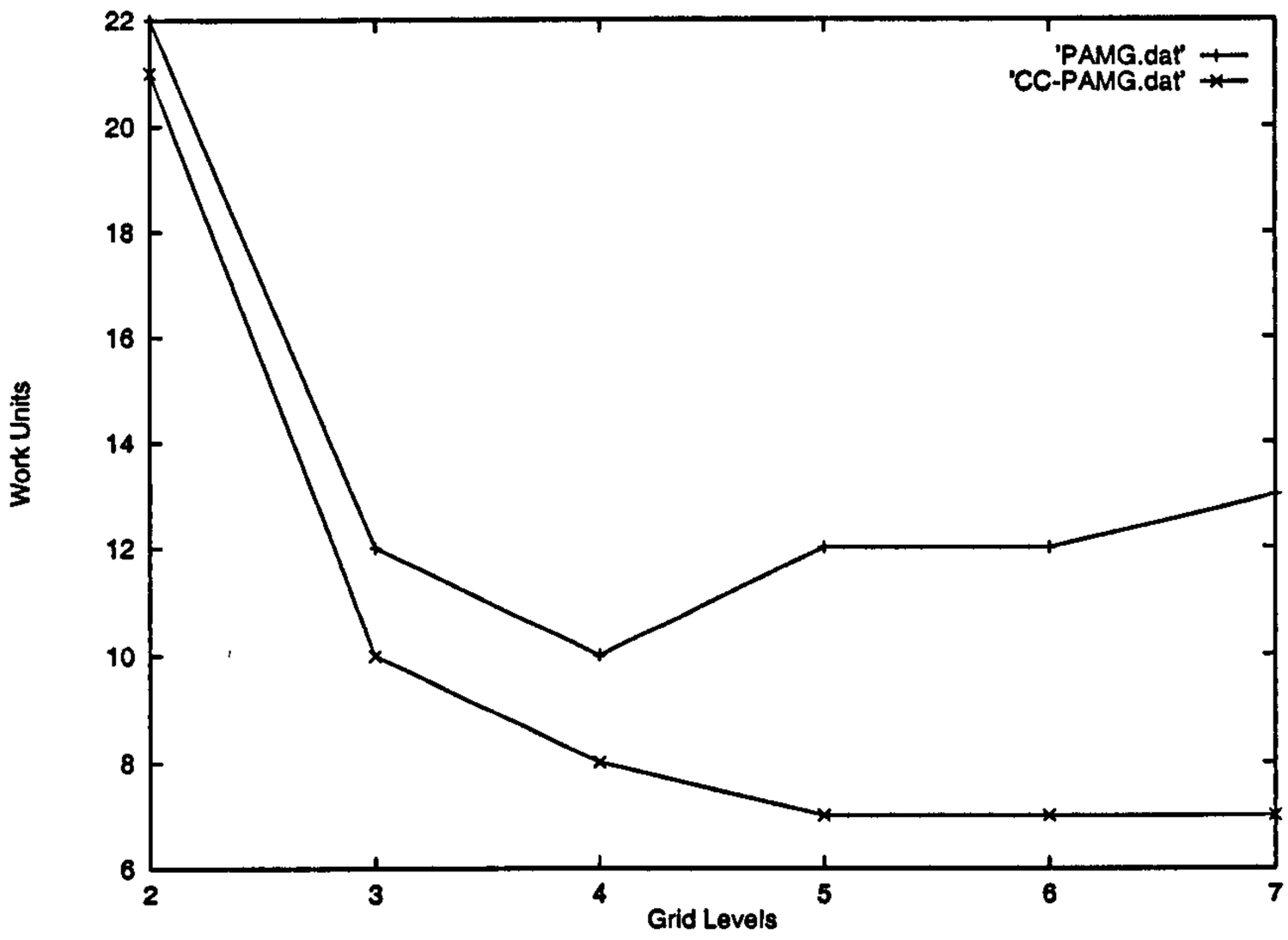


Figure 5.12: Backward Facing Step Problem - Comparison the work units to achieve convergence of the PAMG and CC-PAMG From Level 2 to Level 7, Reynolds Number = 100

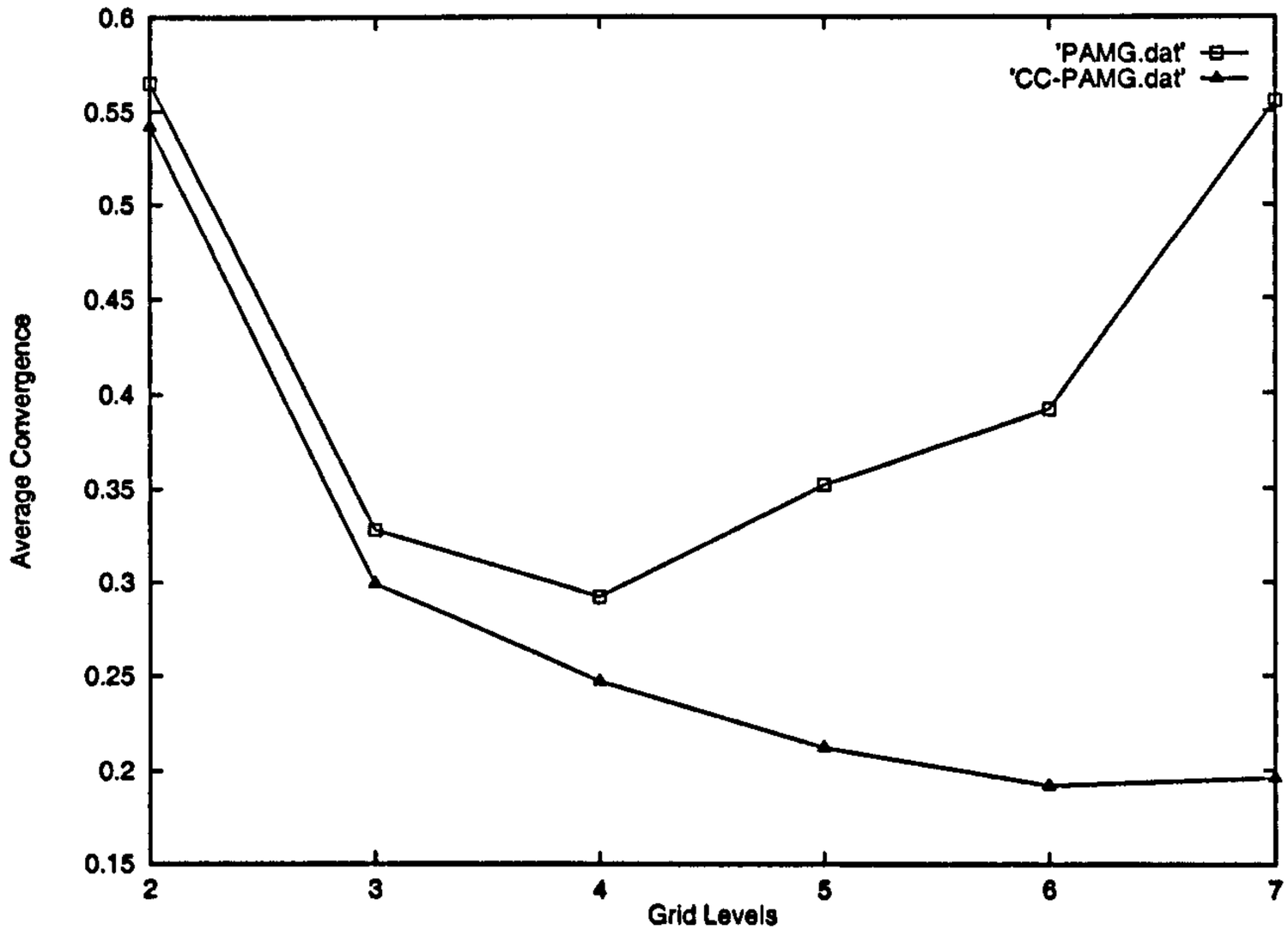


Figure 5.13: Backward Facing Step Problem - Comparison of the PAMG and CC-PAMG convergence ratios on different Adaptive Grid levels, Reynolds Number = 100

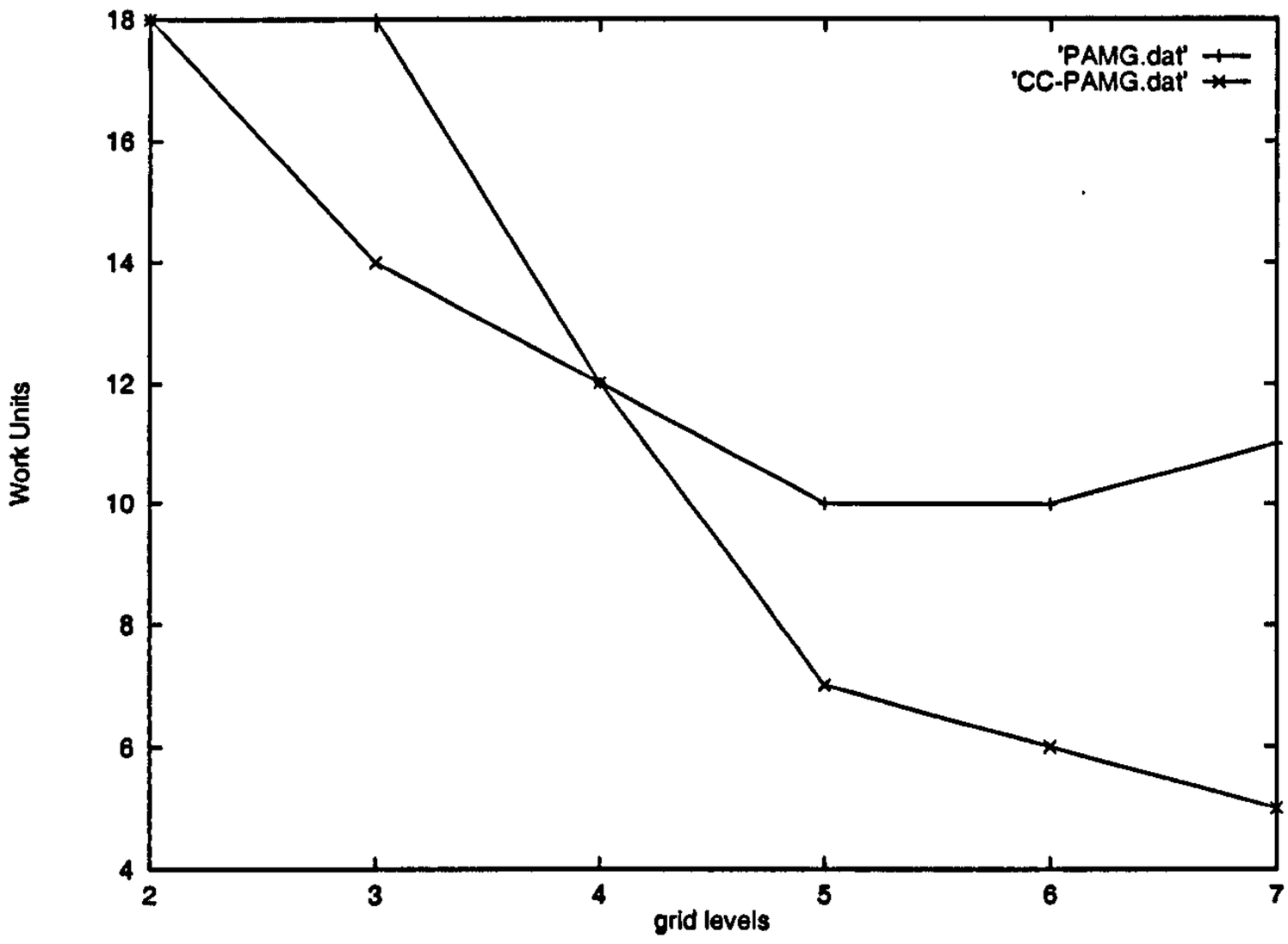


Figure 5.14: Backward Facing Step Problem - Comparison the work units to achieve convergence of the PAMG and CC-PAMG From Level 2 to Level 7, Reynolds Number = 500

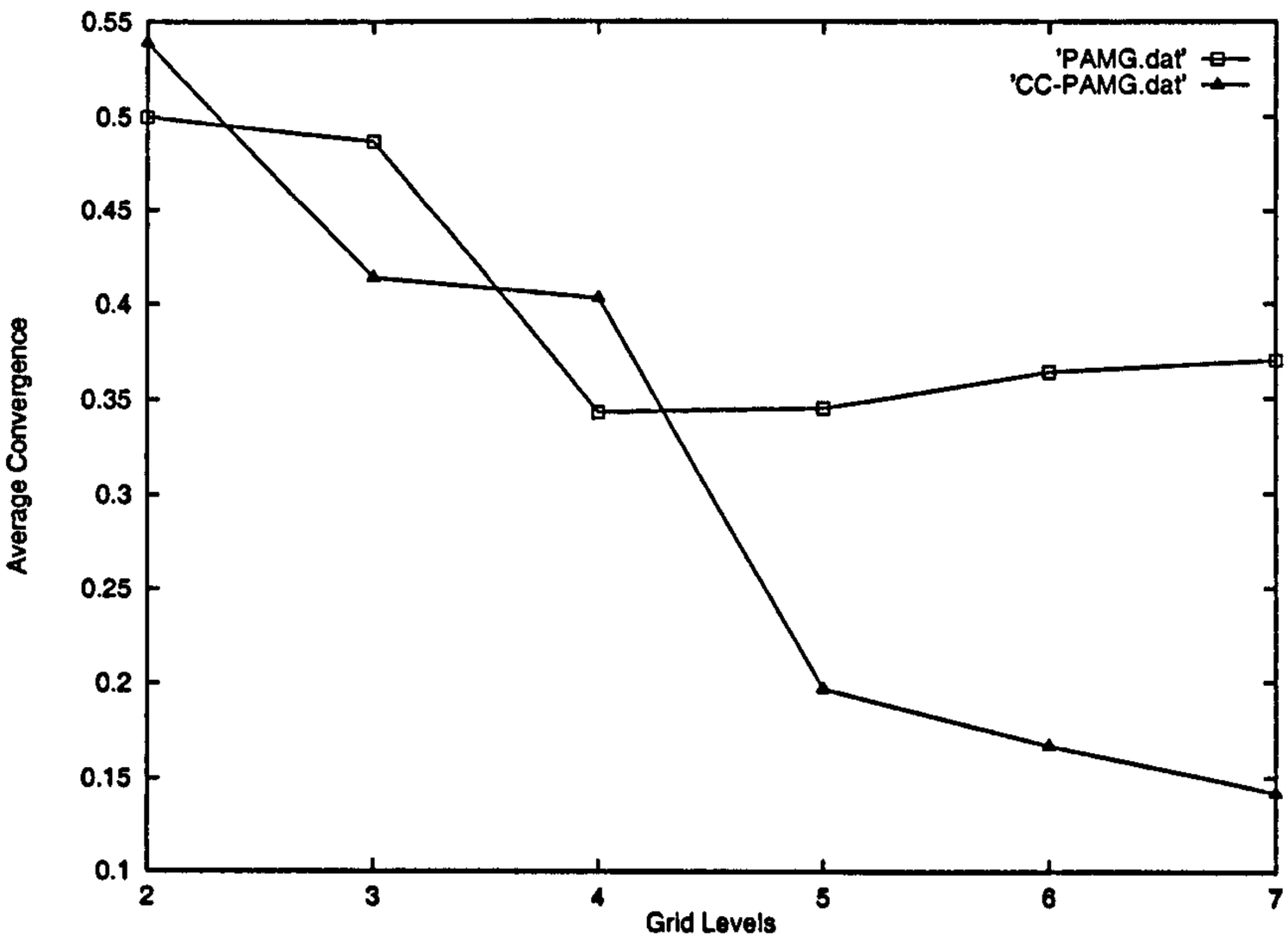


Figure 5.15: Backward Facing Step Problem - Comparison of the PAMG and CC-PAMG convergence ratios on different Adaptive Grid levels, Reynolds Number = 500

Chapter 6

Solution of the Navier-Stokes Equations Using the Cartesian Cut-Cell Method

A method for the calculation of steady incompressible flow was presented in the previous chapter. The procedure, based on a Cartesian cut-cell approach and two-dimensional finite volume scheme, can cope with arbitrarily complex geometries. We discussed the Cartesian cut-cell approach and the error estimate of the cut-cell near the wall using Taylor series. The modification of the near wall treatment of the velocity and pressure were described. In order to assess the accuracy of this new approach which can be used to solve the incompressible flow with arbitrarily complex geometry, several test problems to illustrate the capabilities of the Cartesian cut-cell method are required. This chapter documents the results of incompressible flow computations in several cases and these are compared with some theoretical solutions, and other numerical results.

6.1 Preliminaries

Finite volume algorithms are well developed for structured grids, but with the increased geometric flexibility afforded by unstructured grids, emphasis has recently

been refocused on improving flow solvers for unstructured grids with finite element methods. For structured grids the conservation volumes are described by quadrilaterals occasionally triangles. Normally, unstructured grids use triangular or tetrahedral volumes. In both of these cases, obtaining cell to cell connectivity and cell geometry is simplified since the number of faces/edges for all cell is the same. This simplification in connectivity and geometry also simplifies the flow solver, since there are always a fixed number of geometric entities to be operated on. For example, a three dimensional structured grid always will have six faces neighbours to a cell, accessed by incrementing and decrementing array indices. The complexity and level of effort is switched to the grid generation. The task of grid generation about arbitrarily complicated geometries is a formidable and highly manpower intensive task. It is common that an order of magnitude more time is spent gridding up the volume grid for a calculation about a complicated geometry than is actually spent computing the flow field. The approach presented here tries to overcome this difficulty by taking the user out of the grid generation loop as much as possible, and letting the complexity be switched back to the flow solver. This switch of emphasis now means that more innovative discretization algorithms are needed, since many of the niceties afforded by the simpler grid structures are no longer available.

In this work, the grid is generated recursively from an initial distribution, and during the creation of the grid, the hierarchical relation between newly created cells and their parents is stored in a binary-tree data structure. The cut cells are created automatically using concepts borrowed from computer graphics applications, and since they are hierarchically related to their Cartesian cell 'parents', they can be stored in the linked-list date structure.

This chapter first outlines the solutions for laminar flow along a semi-infinite flat plate from theoretical solution and numerical solutions, results are compared and an error investigation is made. We derive the third-order, nonlinear ordinary differential equations (the Blasius Equation) from the Navier-Stokes equations. We use three different methods to get results: a series analytical solution by using an asymptotic

approach; direct integration of the Blasius equation; and the initial-value problem derived from the Blasius equation which solved by using Runge-Kutta method. We have compared these solutions with the numerical results and validated the new code.

The second test case is the backward facing step problem. Two different Reynolds number flows are investigated in the same physical geometry. The comparisons of the numerical results created by PAMG code and CC-PAMG scheme have been made. The performance of the two solutions is compared and found that CC-PAMG - Cartesian cut-cell algorithm has more rapid convergence based on the new improvements to the grid transfer operators.

Finally, the accuracy of the Cartesian cut-cell algorithm is assessed by performing a grid refinement study for a channel flow which is not aligned with the coordinate axes. The numerical results are compared to the results obtained from theoretical solutions which demonstrate that they have nearly the same results.

6.2 The solutions for laminar flow along a semi-infinite flat plate

The simplest boundary-layer flow to analysis is that associated with steady incompressible flow over a very thin horizontal stationary plate. A uniform fluid having velocity U_∞ flows past a plate which is very thin, so that the streamlines are not displaced by any bluntness of the leading edge. A thin boundary layer forms on the top and only slightly displaces the streamlines.

Figure 6.1 shows the actual experimental streamline pattern of a uniform steady flow of water passing over a thin plate. The plate is 2% thick of the length, with beveled edges. At a Reynolds number of 10,000 (based on the length of the plate), the uniform stream is only slightly disturbed by the thin laminar boundary layer and subsequent laminar wake. The streamlines are made visible by suitably located thin high-resistance wires in the flow. Electric current passing through the wires electrolyzes the water and generates very fine bubbles of hydrogen gas. The hydrogen bubbles are carried along with the water flow and, with proper lighting, make the streamline pattern visible.

Figure 6.2 shows the Blasius boundary-layer profile on a flat plate. The boundary layer slightly displaces the streamlines and gives the flow a small y-component of velocity. The tangential velocity profile in the laminar boundary layer on a flat plate, discovered by Prandtl and calculated accurately by Blasius, is made visible by tellurium. Water is flowing at 9 cm/s. The Reynolds number is 500 (based on distance from the leading edge), and the displacement thickness is about 5 mm. A fine tellurium wire perpendicular to the plate at the left is subjected to an electrical impulse of a few milliseconds duration. A chemical reaction produces a slender colloidal cloud, which drifts with the stream and is photographed a moment later to define the velocity profile.

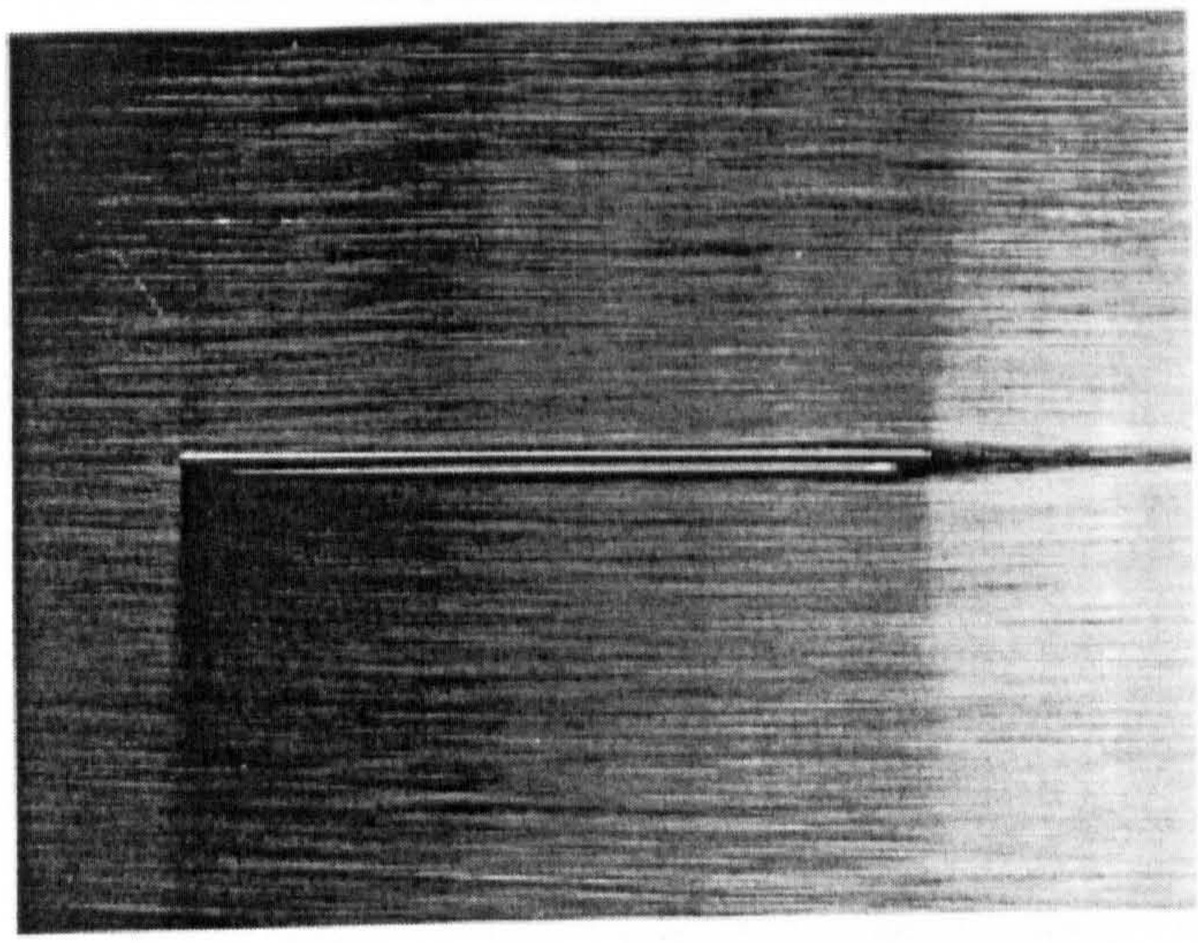


Figure 6.1: Flat plate at zero incidence. ONERA photograph, Werle 1974 [40]

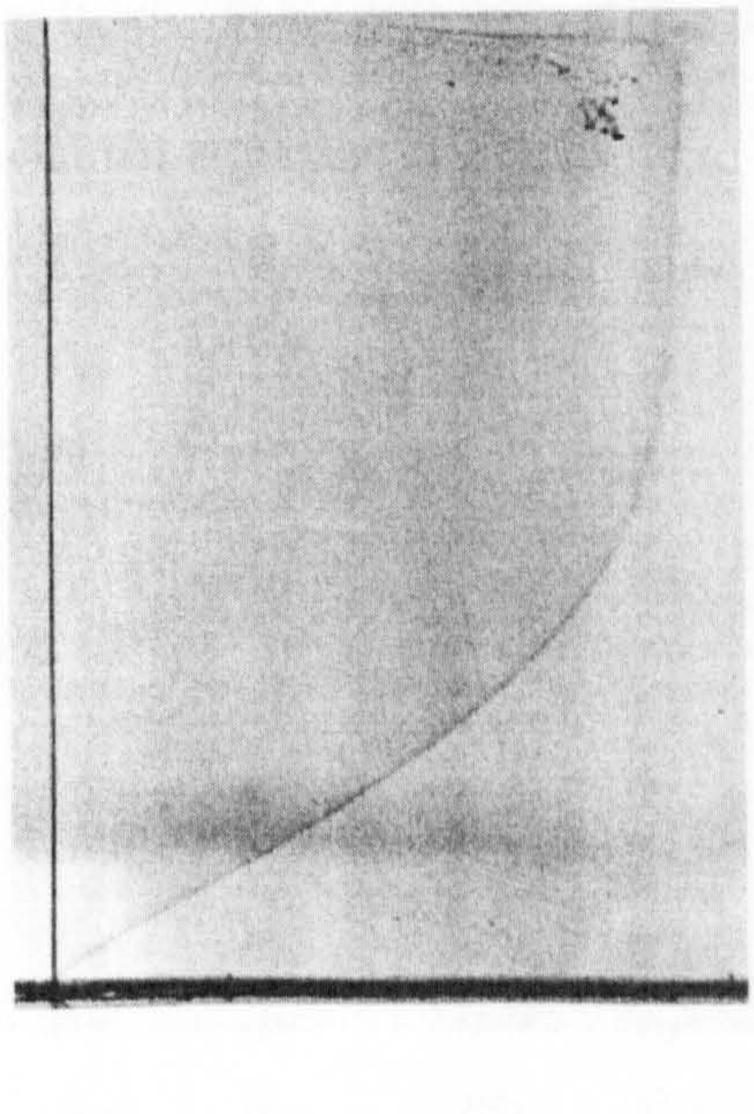


Figure 6.2: Blasius boundary layer profile on a flat plate. Photograph by F. X. Wortmann [40]

6.2.1 The analytical solution for laminar flow along a flat plate

Consider a semi-infinite thin plate with a sharp leading edge immersed in a uniform horizontal incompressible flow in the x-direction with freestream velocity u_∞ (Figure 6.3). For the two dimensional case, the mass equation and Navier-Stokes equations are:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{6.1}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{6.2}$$

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -g - \frac{1}{\rho}\frac{\partial p}{\partial y} + \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{6.3}$$

For uni-directional flow over a flat plate, we take the initial velocity conditions as:

$$u|_{y=0} = 0, \qquad v|_{y=0} = 0, \qquad u|_{y=\infty} = u_\infty.$$

On the boundary $y = 0$, the third equation becomes,

$$\frac{\partial p}{\partial y} = -g\rho$$

The second equation becomes:

$$\frac{\partial p}{\partial x} = 0$$

The Navier-Stokes Equations (6.1) - (6.3) can therefore be written as:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{6.4}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{6.5}$$

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{6.6}$$

With the elimination of pressure as a dependent variable, the problem is over specified. Since v is expected to be very small with respect to u, the last equation may be ignored. Also $\frac{\partial^2 u}{\partial x^2}$ is expected to be much smaller than $\frac{\partial^2 u}{\partial y^2}$ and can therefore be

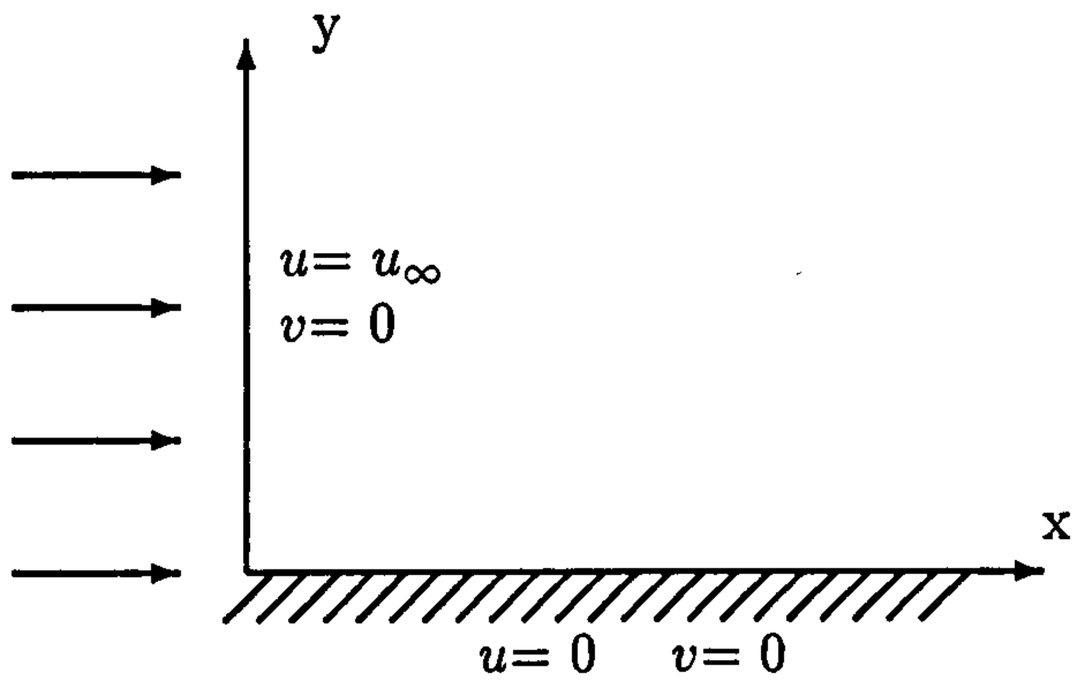


Figure 6.3: Picture to illustrate the flow over a flat plate oriented parallel to the upstream flow.

neglected (more details can be found in H. Schlichting [107]). So the two-dimensional flow equations can be rewritten:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{6.7}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = \nu\left(\frac{\partial^2 u}{\partial y^2}\right) \tag{6.8}$$

A sufficient set of boundary conditions for this model is :

$$u = v = 0, \qquad at \quad y = 0, \quad x > 0,$$

and

$$u \to u_\infty, \qquad as \qquad y \to \infty.$$

We now embark on a procedure which is common to many boundary-layer solutions. Let us transform the independent variables (x,y) into (ξ, η). Introducing the stream function ψ, so the velocities u, v are:

$$u = \frac{\partial \psi}{\partial y}$$

$$v = \frac{-\partial \psi}{\partial x}$$

The boundary conditions are:

$$\frac{\partial \psi}{\partial y} = \frac{\partial \psi}{\partial x} = 0 \tag{6.9}$$

at y = 0, x = 0

$$\frac{\partial \psi}{\partial y} = u_\infty$$

as $y \to \infty$ and for x < 0

$$\psi = 0$$

at y = 0, x > 0

To simplify the calculation, we introduce another two more independent variables ξ, η and where:

$$\xi = x \tag{6.10}$$

$$\eta = \frac{y}{2}\sqrt{\frac{u_{infty}}{\nu x}} \tag{6.11}$$

The factor of '2' is introduced to make the final equation simpler. Introducing a function f(η):

$$f(\eta) = \int_0^\eta \frac{2u(\xi,\eta)}{u_\infty} d\eta, \tag{6.12}$$

from this equation we have:

$$u(\xi,\eta) = \frac{1}{2} u_\infty \frac{df}{d\eta}, \tag{6.13}$$

$$\frac{\partial \psi}{\partial y} = u = \frac{1}{2} u_\infty \frac{df}{d\eta} \tag{6.14}$$

because

$$\frac{\partial \psi}{\partial y} = \frac{\partial \psi}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial \psi}{\partial \eta}\frac{\partial \eta}{\partial y} = \frac{1}{2}\sqrt{\frac{u_\infty}{\nu \xi}}\frac{\partial \psi}{\partial \eta}. \tag{6.15}$$

Therefore

$$\sqrt{\frac{u_\infty}{\nu \xi}}\frac{\partial \psi}{\partial \eta} = \frac{1}{2} u_\infty \frac{df}{d\eta} \tag{6.16}$$

$$\frac{\partial \psi}{\partial \eta} = \frac{1}{2}\frac{\partial}{\partial \eta}\left(\sqrt{u_\infty \nu \xi} f\right) \tag{6.17}$$

Integrating this equation, we have the stream function as follows:

$$\psi = \sqrt{u_\infty \nu \xi} f + c(\xi) \tag{6.18}$$

from the boundary conditions, we have:

$$\left[\frac{\partial \psi}{\partial x}\right]_{y=0} = \left[\frac{\partial \psi}{\partial y}\right]_{y=0} = 0$$

therefore,

$$[\psi]_{y=0} = const$$

$$[\psi]_{\eta=0} = 0, \qquad [f(\eta)]_{\eta=0} = 0$$

therefore

$$c(\xi) = 0$$

$$\psi = \sqrt{u_\infty \nu \xi} f(\eta)$$

The derivatives of the above equations can be transformed into these new variables as

$$\begin{aligned} u = \frac{\partial \psi}{\partial y} = \frac{\partial \psi}{\partial \eta}\frac{\partial \eta}{\partial y} &= \sqrt{u_\infty \nu \xi}\frac{df(\eta)}{d\eta}\frac{\partial \eta}{\partial y} \\ &= \sqrt{u_\infty \nu \xi}\frac{df(\eta)}{d\eta}\frac{1}{2}\sqrt{\frac{u_\infty}{\nu \xi}} \\ &= \frac{1}{2}u_\infty \frac{df(\eta)}{d\eta} \end{aligned} \tag{6.19}$$

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial \eta}\frac{\partial \eta}{\partial x} = \frac{1}{2}u_\infty f''(\eta)\frac{\partial \eta}{\partial x}$$

$$= -\frac{1}{4}u_\infty \frac{\eta}{\xi} f''(\eta) \tag{6.20}$$

$$\frac{\partial u}{\partial y} = \frac{1}{4}u_\infty \sqrt{\frac{u_\infty}{\nu \xi}} f''(\eta) \tag{6.21}$$

$$\frac{\partial^2 u}{\partial y^2} = \frac{u_\infty}{8}\left(\frac{u_\infty}{\nu \xi}\right) f'''(\eta) \tag{6.22}$$

$$v = \frac{1}{2}\sqrt{\frac{u_\infty \nu}{x}}(\eta f'(\eta) - f(\eta)) \tag{6.23}$$

Substituting these expression in the simplified mass and momentum equations, we obtained:

$$f''' + f f'' = 0 \tag{6.24}$$

The boundary conditions are:

$$f = f' = 0, \qquad at \quad \eta = 0$$

$$f' = 1, \qquad at \quad \eta \to \infty$$

This equation is important; it is called Blasius' equation. Blasius first investigated the flat-plate solution using this equation that was the first practical application of Prandtl's boundary-layer hypothesis since its conception in 1904. We started with the partial differential equation for a flat-plate boundary layer problem, transformed both the independent and dependent variables through above equations, and finally obtained an ordinary differential equation for f(η).

Blasius' equation is a third-order, nonlinear, ordinary differential equation. We suppose it has a series solution:

$$f(\eta) = A_0 + A_1\eta + \frac{1}{2!}A_2\eta^2 + \frac{1}{3!}A_3\eta^3 + \quad ... \quad + \frac{1}{n!}A_n\eta^n + \quad ... \tag{6.25}$$

Using the boundary conditions at $\eta = 0$, we have

$$A_0 = A_1 = 0,$$

the series solution can be rewritten as:

$$f(\eta) = \sum_{i=2}^{\infty} \frac{1}{i!}A_i\eta^i$$

Differentiating the series three times, then

$$f''(\eta) = \sum_{i=2}^{\infty} \frac{1}{(i-2)!}A_i\eta^{(i-2)}$$

$$f'''(\eta) = \sum_{i=3}^{\infty} \frac{1}{(i-3)!}A_i\eta^{(i-3)}$$

Substituting it into the Blasius equation shows that:

$$A_3 + A_4\eta + \frac{1}{2!}A_5\eta^2 + \frac{1}{3!}A_6\eta^3 + \quad ... \quad + \quad \left(\frac{1}{2!}A_2\eta^2 + \frac{1}{3!}A_3\eta^3 + \quad ... \right.$$

$$\left. + \frac{1}{n!}A_n\eta^n + ...\right) \quad \left(A_2 + A_3\eta + \frac{1}{2!}A_4\eta^2 + \frac{1}{3!}A_5\eta^3 + \quad ... \right.$$

$$\left. + \frac{1}{n!}A_n + 2\eta^n + \quad ...\right) = 0 \tag{6.26}$$

Expanding, we have:

$$A_3 + A_4\eta + \frac{1}{2!}({A_2}^2 + A_5)\eta^2 + \quad ... = 0 \tag{6.27}$$

This has to be satisfied for any values of η, so we can equate coefficients to obtain:

$$A_3 \quad = A_4 \quad = 0$$

$${A_2}^2 \quad + \quad A_5 \quad = 0$$

...

This procedure can be continued independently, and we obtain:

$$f(\eta) = \frac{A_2}{2!}\eta^2 - \frac{{A_2}^2}{5!}\eta^5 + 11\frac{{A_2}^3}{8!}\eta^8 - 375\frac{{A_2}^4}{11!}\eta^{11}$$

$$+2,7897\frac{{A_2}^5}{14!}\eta^{14} - 3,817,137\frac{{A_2}^6}{17!}\eta^{17} + ... \tag{6.28}$$

Blasius was the first to obtain this series solution and pointed that the range of convergence of the series is finite. The factor A_2 can be determined by the boundary condition $f' = 1$ at $\eta \to \infty$ and numerical calculation. The factor A_2 is:

$$A_2 \cong 0.4696$$

Therefore, the analytical solution of the velocities distribution are:

$$\psi = \sqrt{u_\infty \nu \xi} f \tag{6.29}$$

$$u = u_\infty f'(\eta) \tag{6.30}$$

$$v = \frac{1}{2}\sqrt{\frac{u_\infty \nu}{x}}(\eta f'(\eta) - f(\eta)) \tag{6.31}$$

Integration of the Blasius Equation

It is also possible to integrate the Blasius' equations numerically. In this section the numerical solution of this equation has been concerned with its boundary conditions which in terms of the function F is written as follows:

$$F(0) = F'(0) = 0, \qquad F(\infty) = 1$$

In order to make the process of numerical integration conceptually easier, the Blasius equation has been replaced with three first-order equations. As introduced earlier (Equation 6.12), $F(\eta)$ is the dimensionless stream function. Now let:

$$G(\eta) = F'(\eta)$$

$$H(\eta) = G'(\eta)$$

then the original Blasius equation becomes:

$$H'(\eta) + F(\eta)H(\eta) = 0 \tag{6.32}$$

The above equations now replace the original Blasius equation; the boundary conditions on the new functions G and H are:

$$G(0) = 0, \qquad G(\infty) = 1$$

and, as before, the initial boundary condition of function F,

$$F(0) = 0$$

Integration can proceed in a symbolic way now, for example,

$$G(\eta) = G(0) + \int_0^\eta H(\eta)d\eta \tag{6.33}$$

$$F(\eta) = F(0) + \int_0^\eta G(\eta)d\eta \tag{6.34}$$

from the first two first-order equations. We can integrate the modified Blasius' equation in the same way. The integrated solution of Blasius equation can be written:

$$H(\eta) = H(0)\exp(\int_0^\eta F(\eta)d\eta) \tag{6.35}$$

It is apparent that in all these expressions intermediate values of F, H, and G are needed, however, these data are not available beforehand. Similarly, the term H(0) $\equiv F''(0)$ is not known in advance and must be found as a part of the solution. As in all numerical integrations, the range of integration is divided into discrete steps and the variables F, G, and H are evaluated at discrete locations η_k, $K = 0, 1, 2, ...N$. The variables there are denoted by $F_k, F_{k+1}, G_k, G_{k+1}$, and so on. The spacing between steps will be assumed to be a constant value, $\delta\eta$. Then the discrete analogy to the symbolic equations above are approximately:

$$G_{k+1} = G_k + \frac{(H_{k+1} + H_k)\delta\eta}{2} \tag{6.36}$$

$$F_{k+1} = F_k + \frac{(G_{k+1} + G_k)\delta\eta}{2} \tag{6.37}$$

$$H_{k+1} = H_k - \frac{(F_{k+1}H_{k+1} + F_kH_k)\delta\eta}{2} \tag{6.38}$$

Here again, not all the values at the (k+1) position are available to enable the solution to be stepped from k to k+1. The values must be estimated. Also, H(0) must be estimate and then subsequently improved to achieve the requirement of $G(\infty) = 1$, One possible scheme that is readily implemented is to let:

$$G^*_{k+1} = G_k + H_k\delta\eta$$

then

$$F_{k+1} = F_k + \frac{(G_k + G^*_{k+1})\delta\eta}{2} \tag{6.39}$$

$$H^*_{k+1} = H_k - \frac{F_kH_k\delta\eta}{2}$$

and

$$H_{k+1} = H_k - \frac{(F_kH_k + F_{k+1}H^*_{k+1})\delta\eta}{2} \tag{6.40}$$

Here, the starred quantities are approximate or estimated values of the variables at the next spatial position, which can be used to obtain a correct values in F_{k+1} and H_{k+1}. Then we have:

$$G_{k+1} = G_k + \frac{(H_k + H_{k+1})\delta\eta}{2} \tag{6.41}$$

to complete one cycle of the computation. This cycle is repeated until the maximum value K is reached. From the reference[105], we know that $\eta = 5$ or 6 when the velocity reaches the u_∞. A small step size ($\delta\eta = 0.02$) is chosen, so relatively few iterations are required. The equations described provide one of the simplest examples of the *Runge – Kutta* methods for integration of differential equations. The present methods is accurate through $O(\delta\eta^2)$.

Initial-value problem

A method for solving the Blasius equations, which is very suitable for computers, is to cast the system as an initial-value problem. Suppose we have the transformation:

$$\zeta = k\eta, \qquad \phi(\zeta) = \frac{1}{k} f(\eta)$$

Where k = constant, the Blasius equation becomes:

$$\frac{\partial^3 \phi}{\partial \zeta^3} + \phi \frac{\partial^2 \phi}{\partial \zeta^2} = 0 \tag{6.42}$$

since:

$$(\frac{\partial^2 f}{\partial \eta^2})_{\eta=0} = \alpha$$

then we obtain:

$$(\frac{\partial^2 \phi}{\partial \zeta^2})_{\zeta=0} = \frac{\alpha}{k^3}$$

If we set $k = \alpha_{1/3}$, then:

$$(\frac{\partial^2 \phi}{\partial \zeta^2})_{\zeta=0} = 1$$

Consequently, we pose the following initial-value problem as:

$$\frac{\partial^3 \phi}{\partial \zeta^3} + \phi \frac{\partial^2 \phi}{\partial \zeta^2} = 0 \tag{6.43}$$

$$at \quad \zeta = 0: \quad \phi = \frac{d\phi}{d\zeta} = 0, \qquad \frac{d^2\phi}{d\zeta^2} = 1$$

The initial-value problem was proposed by Weyl [131] and can be solved numerically by a Runge-Kutta method.

η	$f'(\eta)$	$f''(\eta)$
0.0	0.0	0.3321
1.0	0.3298	0.3230
2.0	0.6298	0.2668
3.0	0.8461	0.1614
4.0	0.9555	0.0642
5.0	0.9916	0.0059
6.0	0.9990	0.0024
7.0	0.9999	0.0002
8.0	1.0000	0.0001

Table 6.1: Numerical values of the solution of the Blasius equation (R. H. Sabersky et al. [105])

Before giving the results of the Runge-Kutta method, let us introduce numerical values of the Blasius equation calculated by L. Howarth, who investigated this problem in 1938 (Table 6.1), to test the accuracy of these algorithms in the laminar flow over a very thin flat plate.

Runge-Kutta Method

By using the Runge-Kutta method , here $\phi = \phi(\zeta)$. Suppose the solution at $\zeta_n = nh$, (n = 0, 1, 2, ...,m) is known. Writing:

$$\phi_n = \phi(\zeta_n), \qquad \phi'_n = \phi'(\zeta_n) \qquad \phi''_n = \phi''(\zeta_n)$$

Compute the following quantities:

$$A = -\frac{h^3}{6}\phi_n\phi''_n,$$

$$B = -\frac{h^3}{6}(\phi_n + 0.5h\phi'_n + 0.125h^2\phi''_n + 0.125A)(\phi''_n + \frac{3A}{h^2}),$$

$$C = -\frac{h^3}{6}(\phi_n + 0.5h\phi'_n + 0.125h^2\phi''_n + 0.125A)(\phi''_n + \frac{3B}{h^2}),$$

$$D = -\frac{h^3}{6}(\phi_n + h\phi'_n + 0.5h^2\phi''_n + C)(\phi''_n + \frac{6C}{h^2}).$$

Then

$$\phi_{n+1} = \phi_n + h\phi'_n + \frac{h^2}{2}\phi''_n + \frac{1}{20}(9A + 6B + 6C - D), \tag{6.44}$$

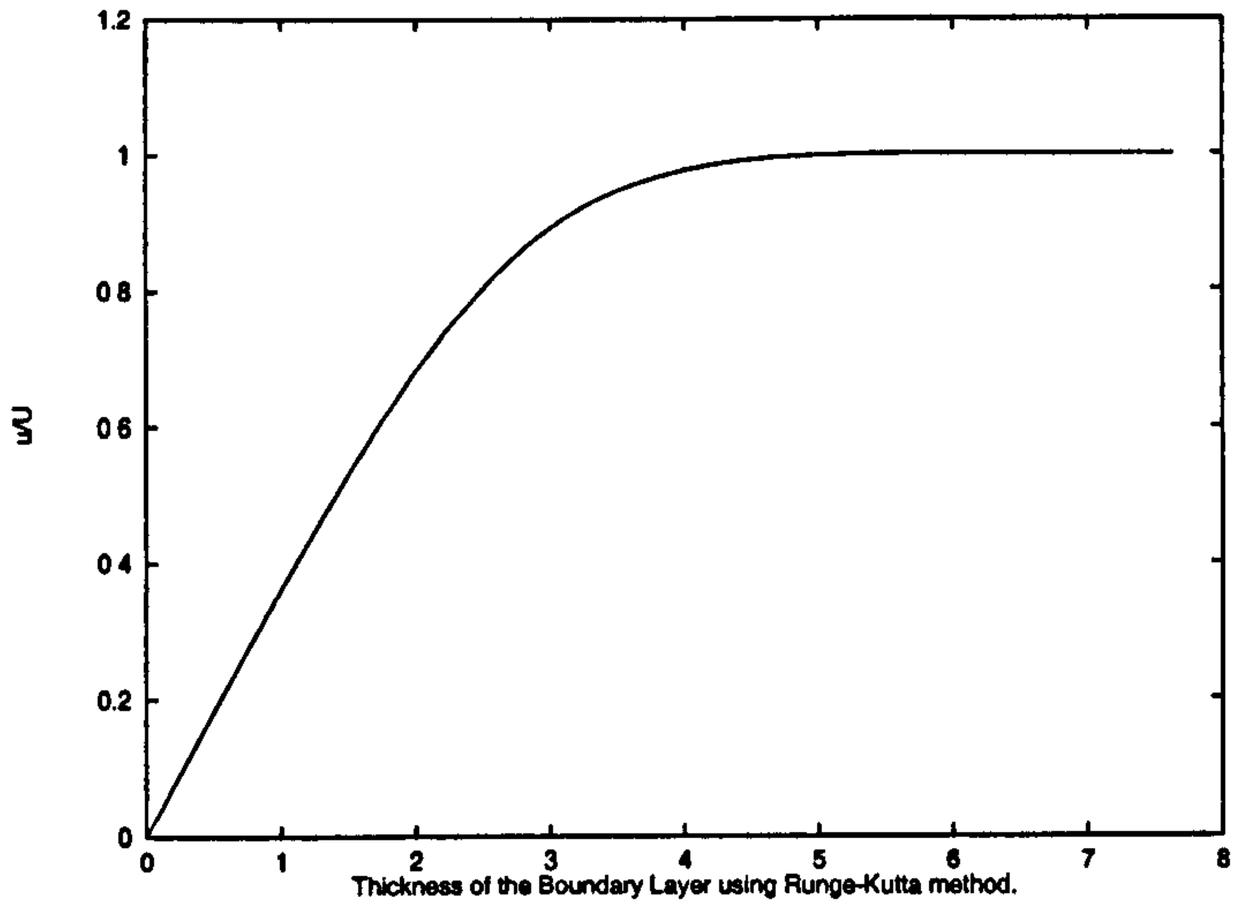
$$\phi'_{n+1} = \phi'_n + h\phi''_n + \frac{1}{h}(A + B + C), \tag{6.45}$$

$$\phi''_{n+1} = \phi''_n + \frac{1}{h^2}(A + 2B + 2C + D). \tag{6.46}$$

When the above inequality is satisfied the equation $\mid \phi'_{n+1} - \phi'_n \mid$ less than ϵ, then stop calculations, otherwise proceed one more steps and repeat the procedure.

In order to compare the different numerical results, a code for Runge-Kutta Method has been developed to solve this particular problem. The numerical solutions of Blasius equation using Runge-Kutta Method are give in Table 6.2 and is plotted in Figure (6.4) in the form of $f'(\eta) = \frac{u}{u_\infty}$ as a function of η; Figure (6.4) also shows the solution of $f''(\eta)$ as a function of η. Note that this curve is the velocity ratio profile and that it is a function of η only, but η is function of x and y ($\eta = \frac{y}{2}\sqrt{\frac{u_\infty}{\nu\xi}}$). Consider two different x stations along the plate. In general, u = u(x,y) and the velocity profiles in terms of u=u(y) at given x locations will be different. Clearly, the variation of u normal to the wall will change as the flow progresses downstream. However, when plotted versus η, we see that the profile $u = u(\eta)$ is the same for all x stations.

From Figure (6.4), we found that the velocity profile of the Runge-Kutta solution is a smooth curve. Initially it increases linearly. After the point $\eta = 3$, the curve increases slowly and finally approaches a horizontal line. We find that the velocity $u/u_\infty = F'(\eta)$ is correctly approximated with the exponential functions. Actually, when $\eta = 5$, it has nearly reached this value. For the acceleration profile, it starts at a certain acceleration and gradually tends to 2% of the original acceleration at the position $\eta = 5$. It has the S -shape to deceleration profile.



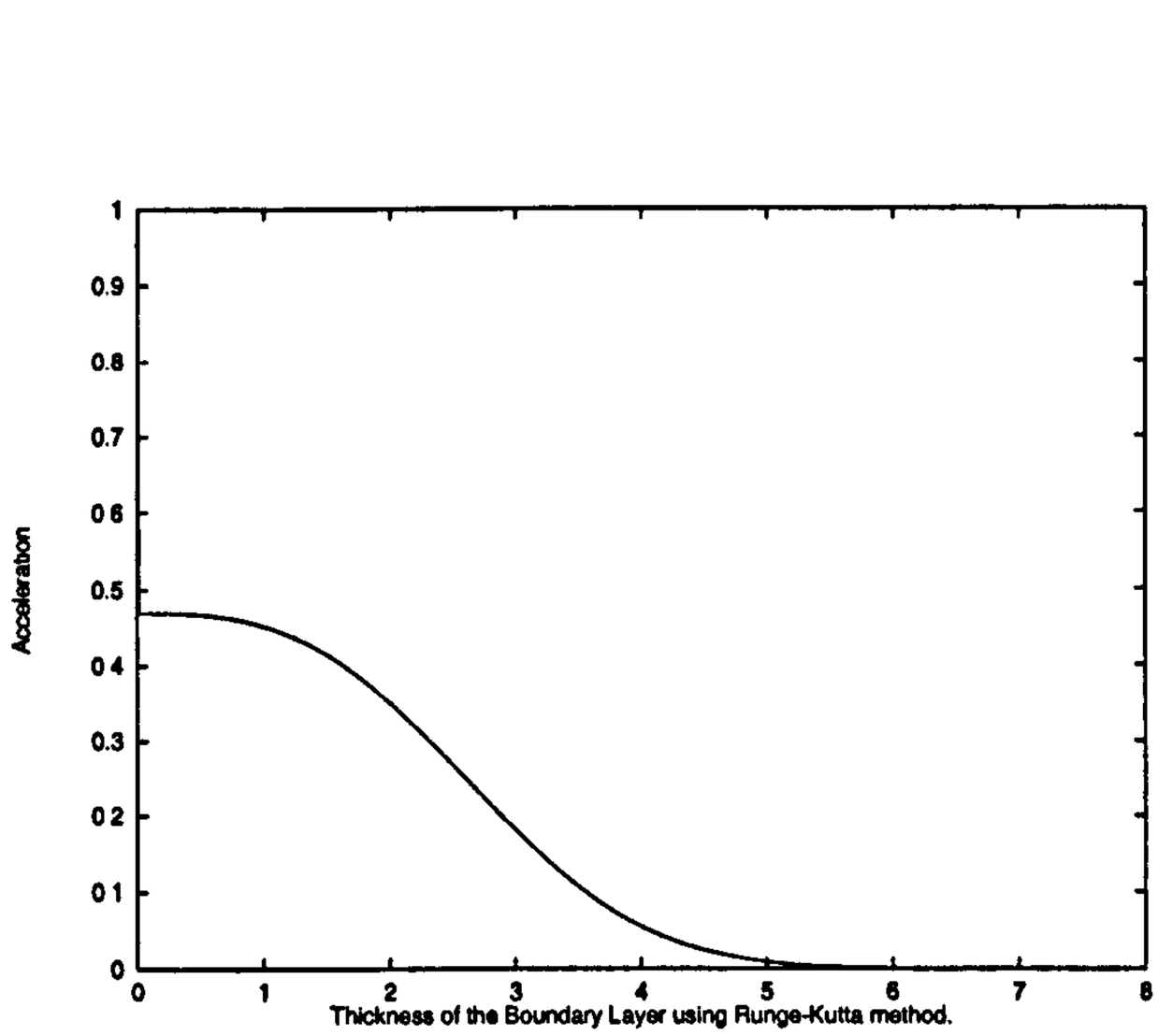


Figure 6.4: Flat Plate boundary layer velocity and acceleration profile corresponding to the equation $F''' + FF'' = 0$.

η	$f(\eta)$	velocity($f'(\eta)$)	$f''(\eta)$
initial data :	0.0000	0.0000	1.0000
0.017	4.999999E-05	6.041913E-03	4.695999E-01
1.001	1.823418E-01	3.622015E-01	4.526143E-01
2.003	7.116163E-01	6.817243E-01	3.510107E-01
3.004	1.509611E-00	8.897915E-01	1.811651E-01
3.997	2.443937E-00	9.756834E-01	5.556500E-02
4.998	3.434483E-00	9.968939E-01	9.396724E-03
6.000	4.434696E-00	9.998153E-01	8.694202E-04
7.001	5.436011E-00	1.000040E-00	4.390146E-05
7.498	5.932565E-00	1.000048E-00	7.977789E-06
7.638	6.073256E-00	1.000049E-00	4.789442E-06
8.001	6.073556E-00	1.000050E-00	4.789542E-06

Table 6.2: Numerical values of the solution of the Blasius equation using Runge-Kutta Method

6.2.2 Numerical solution by solving the complete equation

The laminar incompressible flow over a flat plate is a classic fluid dynamics problem. However, no exact analytical solution exist. A flat plate at zero incidence is a simple geometry. Herein lies the real benefit of CFD. Traditionally, a boundary-layer solution technique has been used to solve this problem as discussed in the above section. Although results obtained from boundary-layer techniques are reasonably good for certain applications, there are limits to the classes of problems which can be solved especially in complex geometries. In this section, we do not introduce any variables to simplify the Navier-Stokes equations, instead a novel method has been used to solve the full discretised Navier-Stokes equations. The solutions for laminar flow along a flat plate obtained with four different methods are compared and the efficiency of the Cartesian grid approach method has been investigated.

Description of the Navier-Stokes equation

The Navier-Stokes equations are discretized using a hybrid scheme and a finite volume method which employs second-order central differences in the limit. The schemes provide reasonable accuracy for sufficiently small mesh sizes on coarse grids. A standard staggered grid is overlaid on the domain. The velocities are associated with the cell faces and the pressure is associated with the cell centres. The grid is defined with dimension spacing Δx and Δy. If we consider the (i,j) -th cell then the pressure associated with the cell centre is denoted by p_{ij}, the x -component of the velocity associated with the centre of the right-hand face is denoted by $u_{i+1/2j}$ and the y-component is denoted by $u_{ij+1/2}$. The finite-difference equations can be written as in Chapter 3:

$$A_c^u u_{i+1/2,j} = A_n^u u_{i+1/2,j+1} + A_s^u u_{i+1/2,j-1} + A_e^u u_{i+3/2,j} + A_w^u u_{i-1/2,j}$$

$$+\frac{p_{i,j} - p_{i+1,j}}{\Delta x} \tag{6.47}$$

$$A_c^v v_{i,j+1/2} = A_n^v v_{i,j+3/2} + A_s^v v_{i,j-1/2} + A_e^v v_{i+1,j+1/2} + A_w^v v_{i-1,j+1/2}$$

$$+\frac{p_{i,j} - p_{i,j+1}}{\Delta x} \tag{6.48}$$

$$\frac{u_{i+1/2,j} - u_{i-1/2,j}}{\Delta x} + \frac{v_{i,j+1/2} - v_{i,j-1/2}}{\Delta y} = 0 \tag{6.49}$$

while the coefficients are defined as follows. Letting $\phi = u, v$, we have:

$$A_c^\phi = A_n^\phi + A_s^\phi + A_e^\phi + A_w^\phi$$

$$A_w^\phi = max(|C_{x-}^\phi|, D_{x-}^\phi) + C_{x-}^\phi$$

$$A_e^\phi = max(|C_{x+}^\phi|, D_{x+}^\phi) - C_{x+}^\phi$$

$$A_s^\phi = max(|C_{y-}^\phi|, D_{y-}^\phi) + C_{y-}^\phi$$

$$A_n^\phi = max(|C_{y+}^\phi|, D_{y+}^\phi) - C_{y+}^\phi$$

where the differential form of the coefficients is used to give the correct scalings across the grids.

6.2.3 Numerical solutions of the flat plate flow

In the above subsection, we have obtained the discretized Navier-Stokes equation. The numerical computation is made with a free-stream taken to be $u_\infty = 1.8$. For simplicity, the reference length is taken so that $0 \leq x \leq 12.5$ and the Reynolds number is chosen to be 200. The computation domain and boundary conditions are chosen as follows:

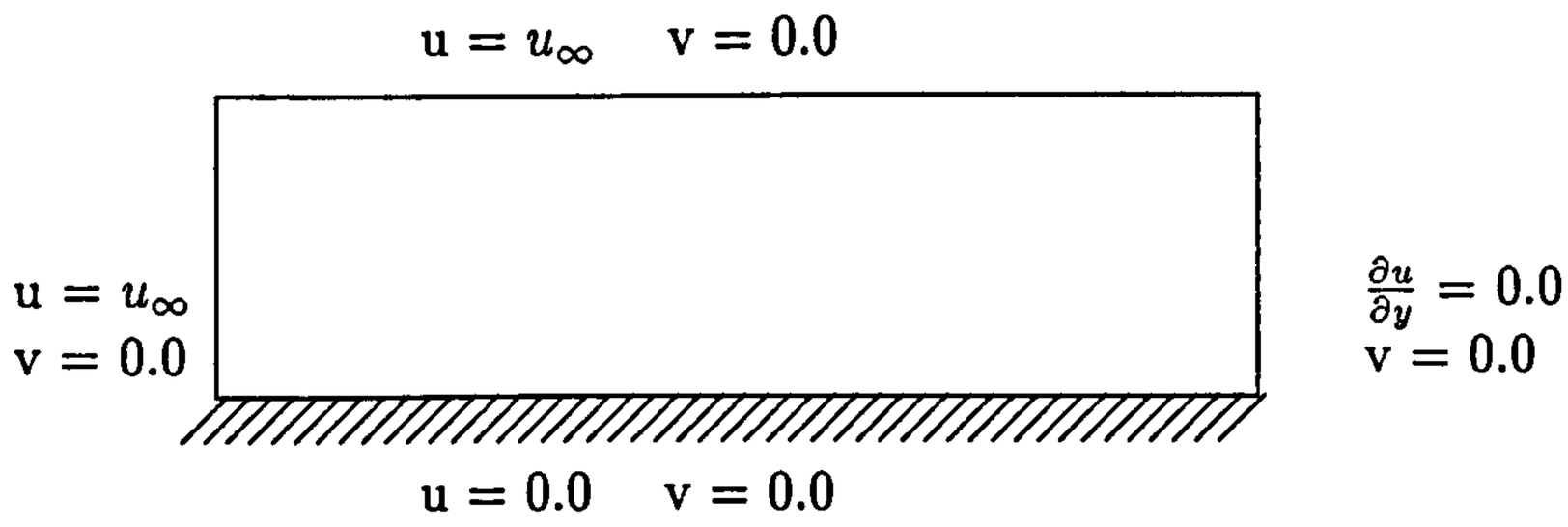


Figure 6.5: Geometrical representation of the flat plate flow.

At the upper, outflow and stagnation streamline boundaries the velocity components are extrapolated from the interior and the pressure is specified at its reference value. The origin cell of the grid is located so that its south face lies along the plate surface. The meshes for this example were generated by using the Cartesian-cell grid. Five level meshes are used, the finest having a few divisions located the front edge of the flat plate and the coarsest grid contains 400 cells illustrating the whole computation domain. Figure 6.6 shows the streamline and pressure profiles of the laminar flow along a semi-infinite flat plate. Figure 6.7 shows the rectangular meshes of the cell construction using the original PAMG code. Figure 6.8 shows the rectangular meshes of the cell constructed using the CC-PAMG code. In this case, it is found that near the boundary further refinement occurs so the solution of the boundary layer is much more accurate than with the original PAMG solutions. The two different velocity profiles to the flat plate located at x = 7.0 are shown in Figure 6.9 that CC-PAMG solution has a good agreement with the Blasius solution, especially with large Reynolds number.

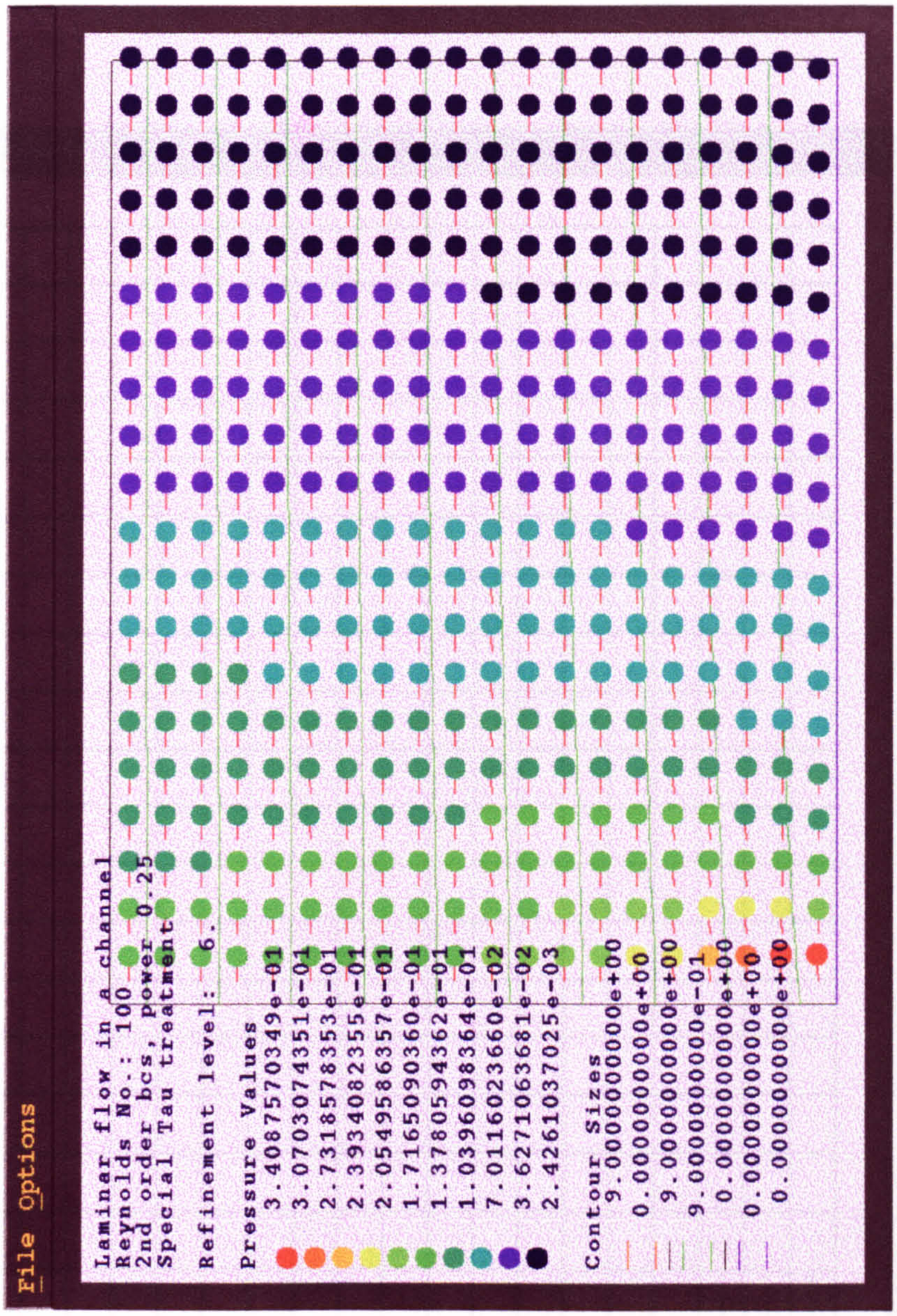


Figure 6.6: Streamlines and pressure solutions for laminar flow along a Flat Plate. The biggest corresponding pressure contour is located in the leading edge of the flat plate.

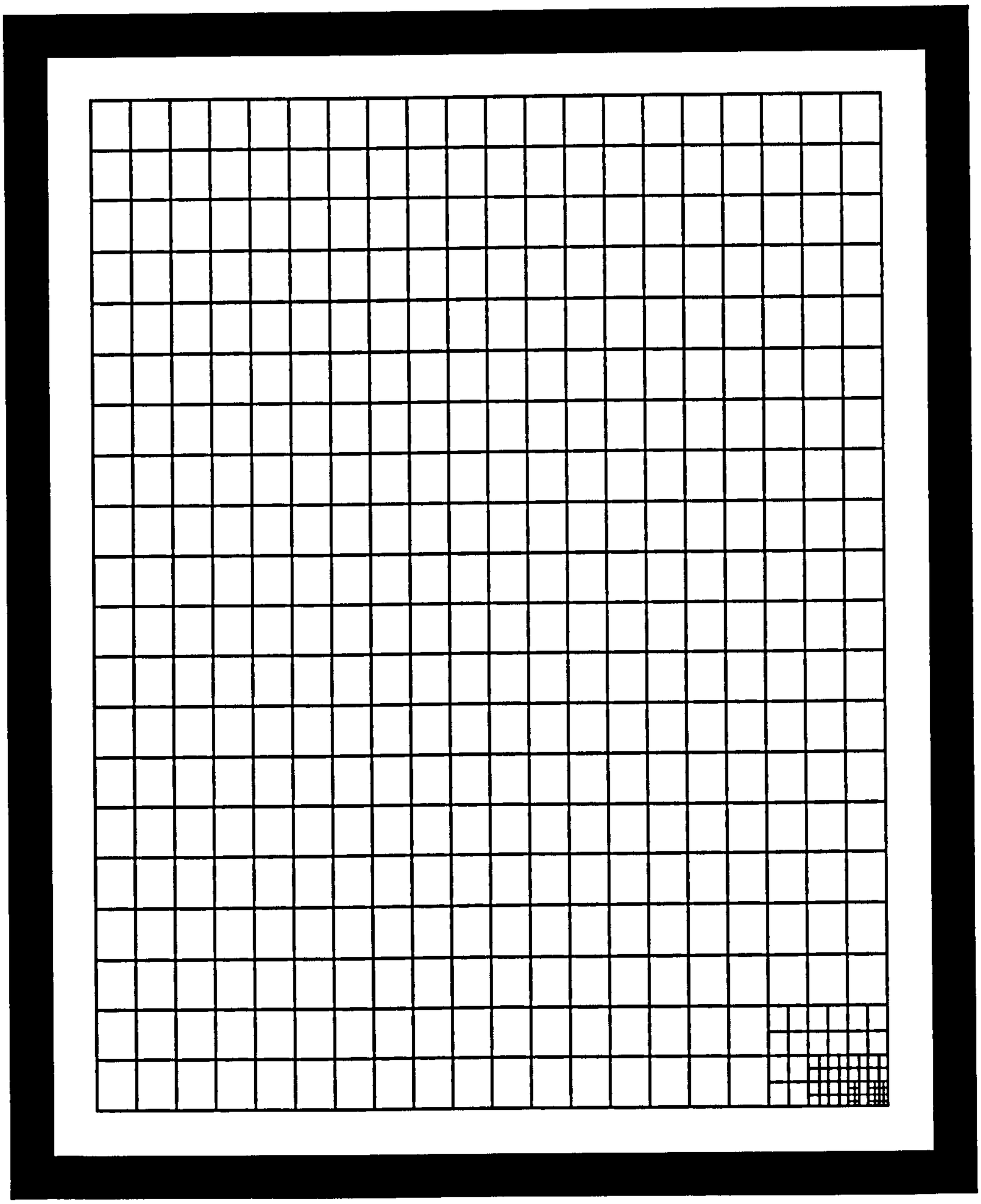


Figure 6.7: Final Adapted Grid for flat plate flow - Original PAMG Solution, only the leading edge has been refined adaptively due to the pressure contour and velocity changes.

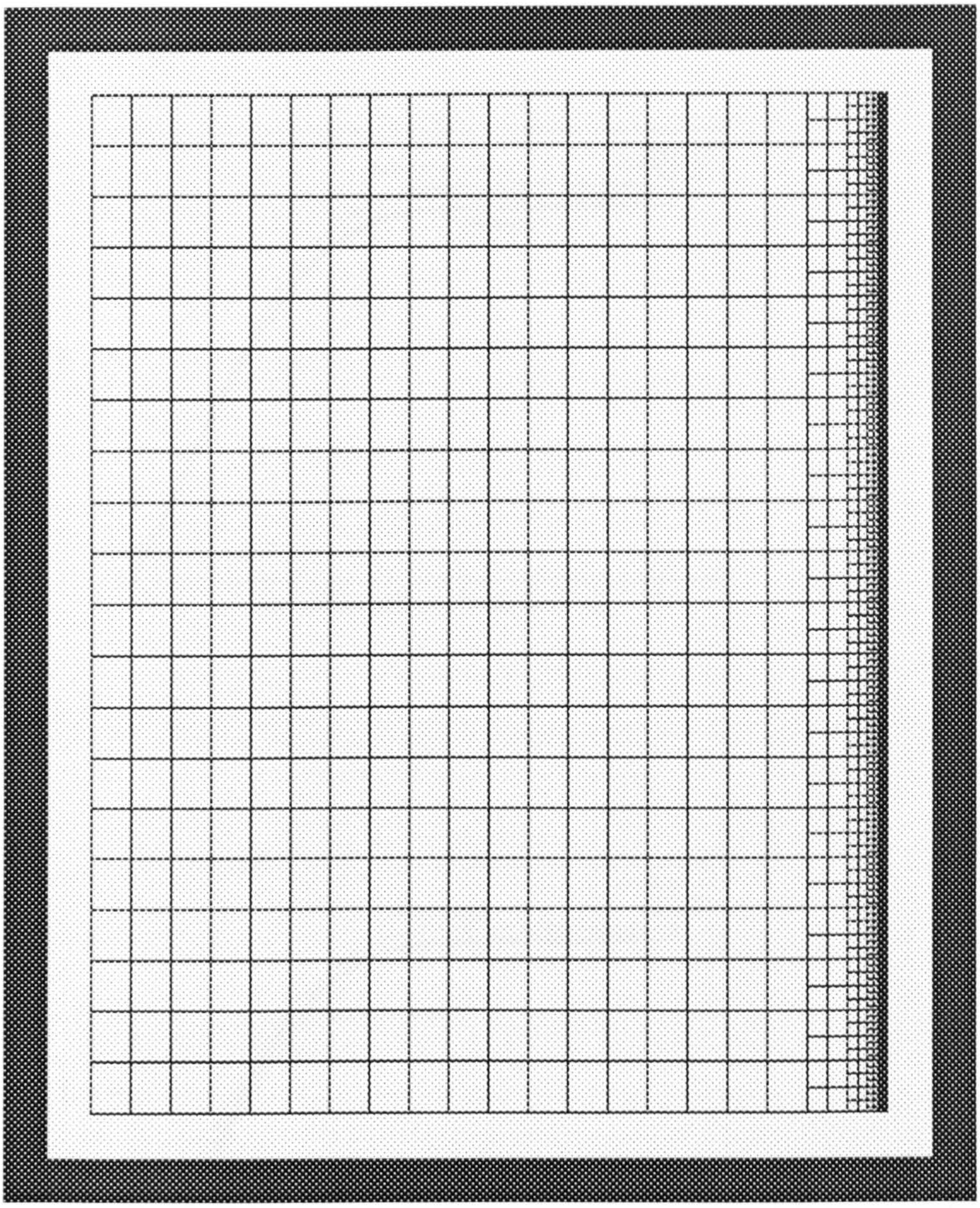
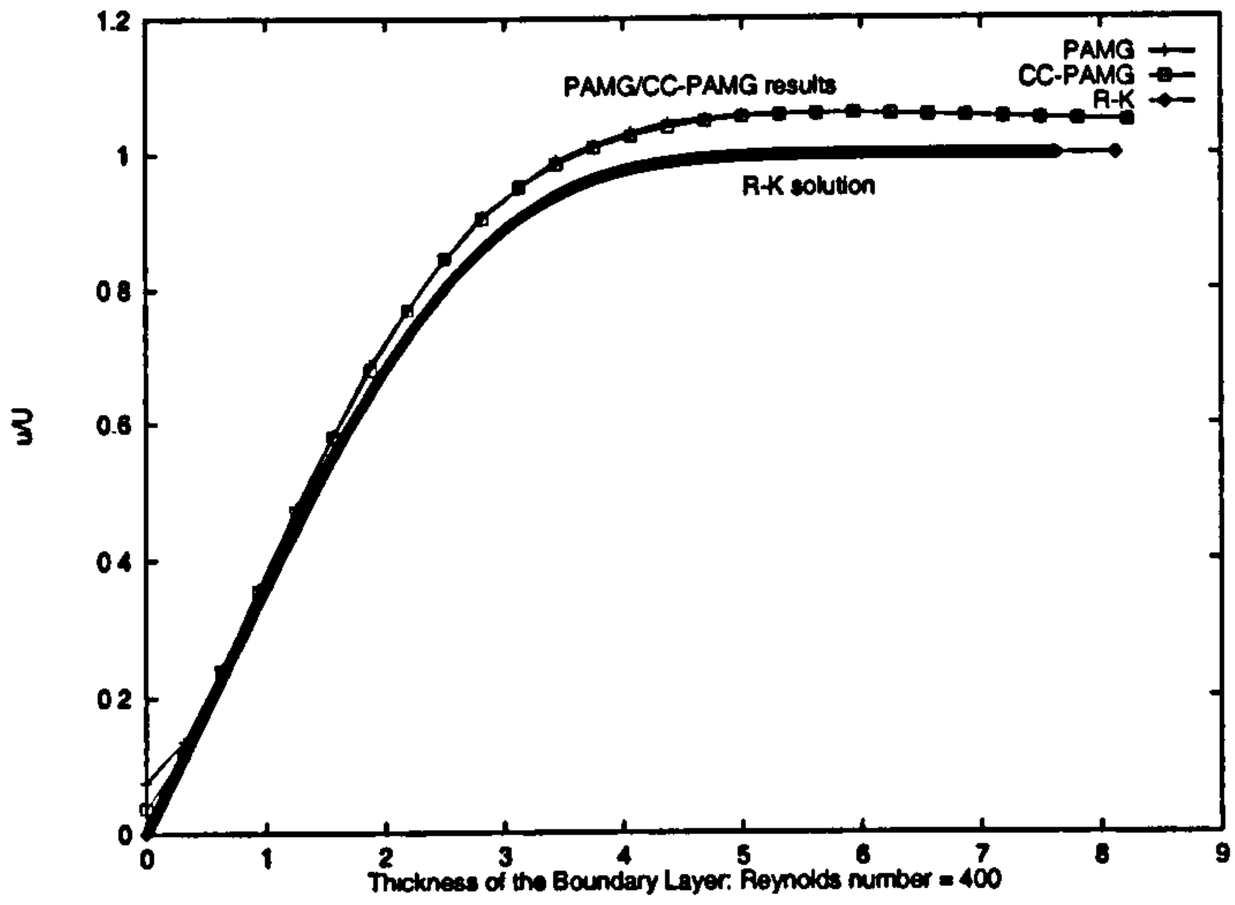


Figure 6.8: Final Adapted Grid for flat plate flow- CC-PAMG Solution, they are refinement near the boundary positions to deal with the boundary layers comparing to other numerical solutions.



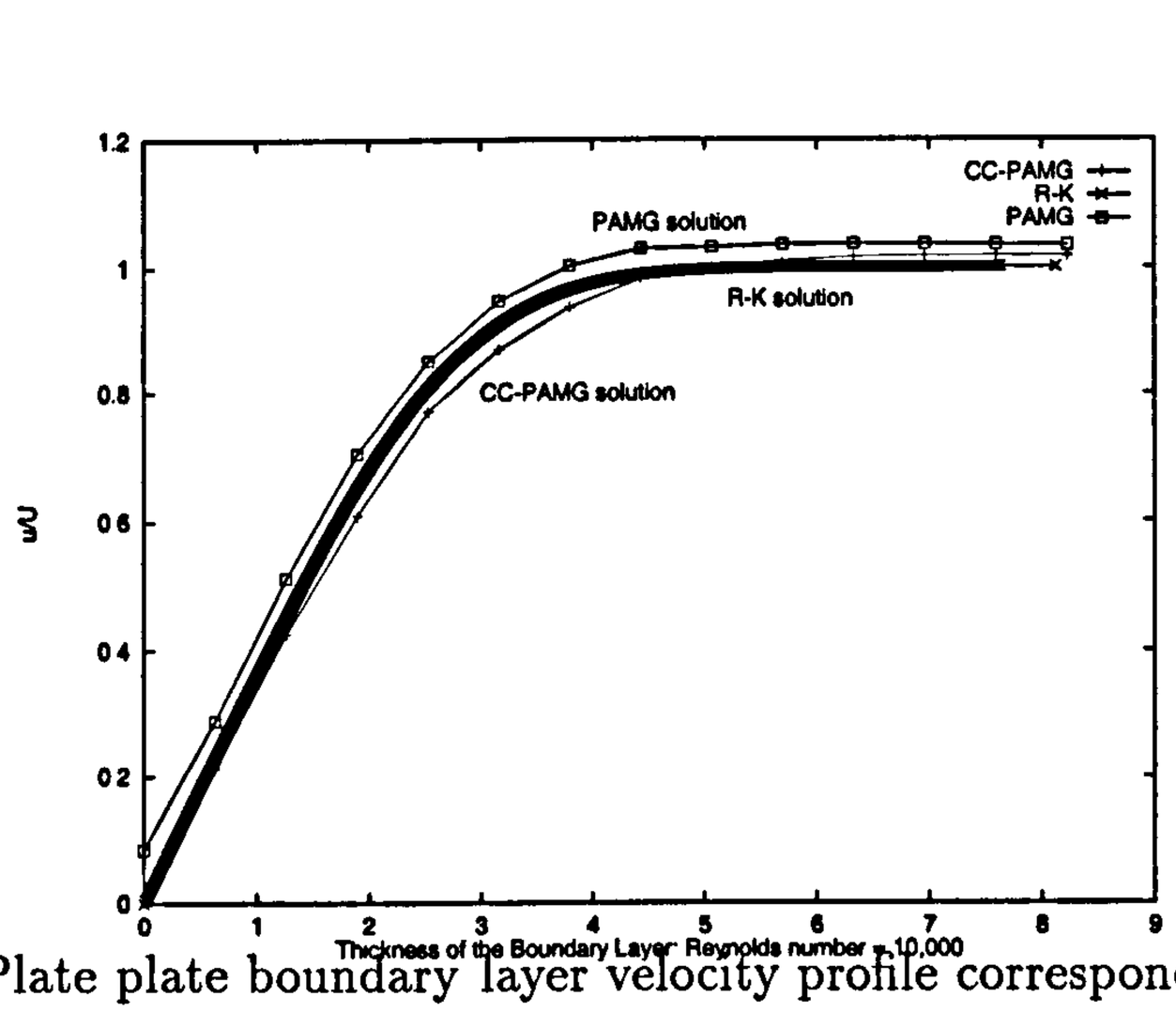


Figure 6.9: Plate plate boundary layer velocity profile corresponding to the Navier-Stokes equations for two different Reynolds numbers, compared with Runge-Kutta solutions

Figure 6.9 shows the two different velocities of PAMG and CC-PAMG compared with Runge-Kutta solutions. Near the wall boundary, the original PAMG solution has bigger errors in the boundary layer, while the CC-PAMG solution has small errors. The reason is that the original PAMG code uses bilinear interpolation to calculate the required velocities which causes big errors along the boundary layer. The refinement only occurs in the entrance part so that the errors along the boundary layer cannot be reduced by the adaptive refinement algorithm. We have considered this weakness of the bilinear interpolation method and modified it to deal with the boundary layer problems.

Y Distance	η	U velocity	V velocity	Pressure
0.0000	0.0000	0.000000	0.000000	0.001381
0.0127	1.2658	0.212756	0.000270	0.002763
0.0253	2.5316	0.425167	0.000780	0.002766
0.0380	3.7975	0.609364	0.001577	0.002769
0.0506	5.0633	0.770483	0.002431	0.002773
0.0633	6.3291	0.870637	0.003181	0.002777
0.0759	7.5949	0.937933	0.003745	0.002783
0.1013	10.1266	0.999088	0.004289	0.002795
0.1266	12.6582	1.016064	0.004447	0.002808
0.1519	15.1899	1.018243	0.004484	0.002821
0.1772	17.7215	1.017966	0.004483	0.002834
0.2025	20.2532	1.017406	0.004479	0.002846

Table 6.3: CC-PAMG Numerical values of the solution of the Navier-Stokes equation (x = 7.0); Reynolds = 10,000

In the low Reynolds number regime and using the inlet (u_∞) profile in this cases, the velocity near the boundary layer is relatively small and can be neglected for this situation. For the laminar flow along the flat plate, the boundary layer is the key

point to investigate. At low Reynolds numbers, the errors are increased in the calculated field, the velocity profiles are bigger than the maximum velocity profiles. When the Reynolds number increases, the velocity profiles are approached to the maximum velocity. At higher Reynolds number, the CC-PAMG algorithm simulates the plate flow very well. The velocity profiles are displayed in the second figure in Figure 6.9 and the solution data are shown in Table (6.3).

Error investigation and Discussion

In the above studies, we have derived the ordinary differential equations from the Navier-Stokes equation with some simplification. In order to solve these ordinary differential equations, we introduced three kinds of methods: integration of the ordinary differential equations with some coefficient assumptions; the Blasius method with a power series solution; and finally the Runge-Kutta numerical method. Then we introduced our new adaptive multigrid, finite-volume solution procedure for Navier-Stokes equation using the Cartesian cell approach.

η	Runge-Kutta ($\frac{u}{U_\infty}$)	PAMG ($\frac{u}{U_\infty}$)	CC-PAMG ($\frac{u}{U_\infty}$)	Err_1	Err_2
0.0000	0.000000	0.000000	0.000000	0.0443	0.0081
0.6329	0.229069	0.287314	0.212756	0.0562	0.0163
1.2658	0.453782	0.511643	0.425167	0.0578	0.0286
2.5362	0.808545	0.850678	0.770483	0.0421	0.0380
3.1645	0.910376	0.946864	0.870637	0.0365	0.0397
4.4304	0.989231	1.029625	0.981703	0.0404	0.0075
6.9601	1.000038	1.046023	1.017551	0.0459	0.0175
8.2278	1.000048	1.043966	1.017792	0.0431	0.0177
9.4956	1.000049	1.041947	1.017686	0.0419	0.0176

Table 6.4: Comparison of error analyses with three different methods, where Err_1 = Runge-Kutta – CC-PAMG and Err_2 = Runge-Kutta – PAMG

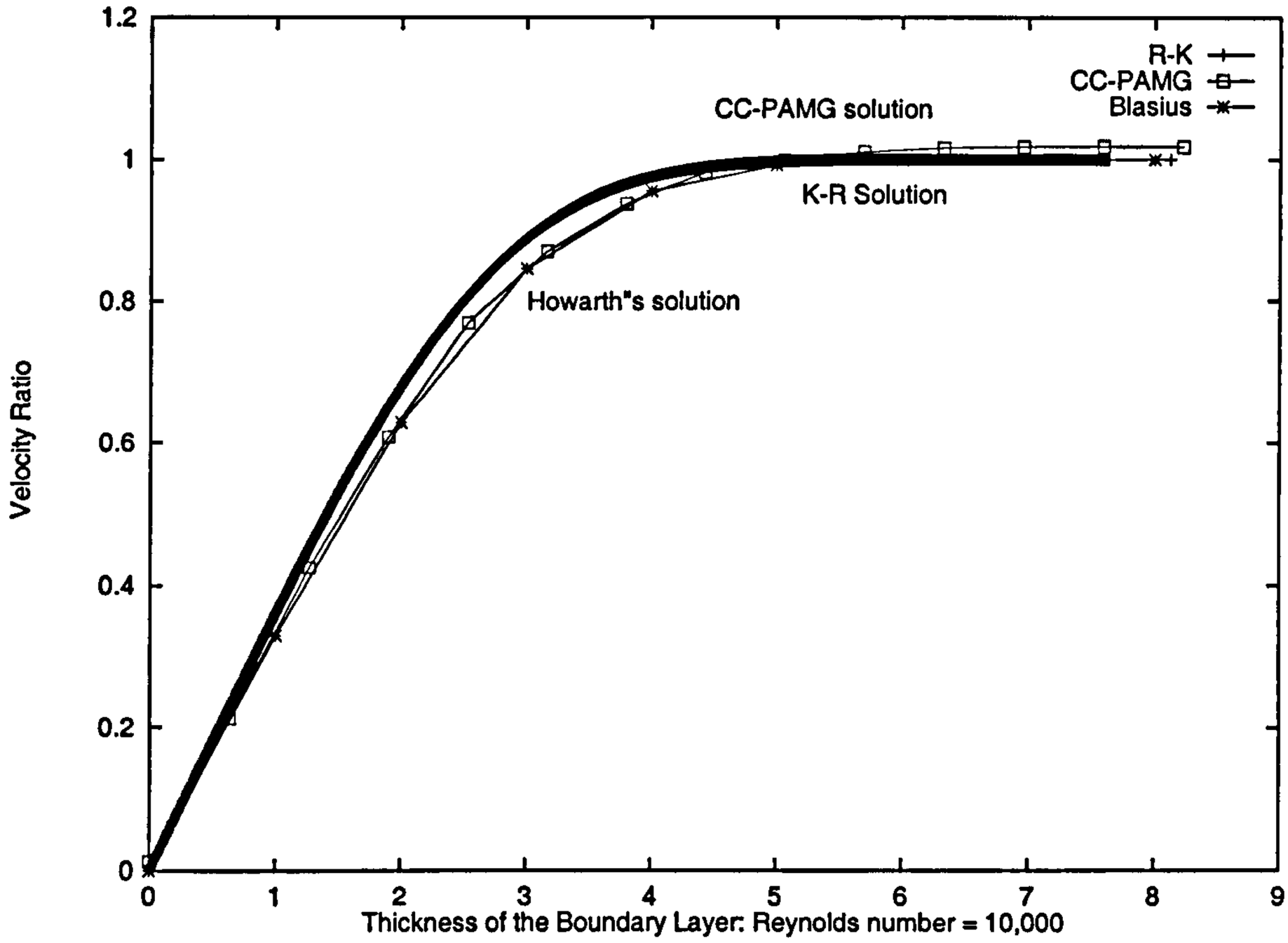


Figure 6.10: Velocity Profile corresponding to the Three Difference Methods

Now we have four different results for the laminar incompressible flow over a flat plate. To compare these results, we first combined these solutions together, found out the difference and then made the error investigation.

Table 6.4 shows the error analysis with the three different methods. The first column gives the η value, and the second column gives the solution obtained using the fourth-order Runge-Kutta method. The third and fourth columns are the numerical solution of Navier-Stokes equations obtained using the CC-PAMG and PAMG schemes respectively. Column error_1 is the difference between Runge-Kutta method and the PAMG solutions, its maximum errors occurred in $\eta = 0.63$, and is 0.058. But the difference between the Runge-Kutta method and CC-PAMG algorithm is small. These errors are displayed in column error_2. Figure 6.10 illustrates the velocity ratios with three different solution.In both cases we see the improvements which the new higher-order interpolation methods makes to the CC-PAMG results.

Conclusion

The above test case demonstrates the complete Navier-Stokes solution for laminar incompressible flow over a flat plate. A flat plate, at zero incidence, is a simple geometry, yet the solution unveils an amount of interesting physics. We have used this section to describe new areas of the CC-PAMG code and to validate these improvements vis-a-vis the original PAMG algorithm. Runge-Kutta method yields much better results since it is 4-order accuracy. However, some simplifications for the calculation were made and higher-order items were ignored.

Comparing with the other three solutions, the results of CC-PAMG agree well with theory and other numerical solutions. Although the velocity is not predicted as well as in the theory after $\eta = 5.0$, the error is less than 2 % in this position. The improvements compared to the PAMG code are nearly two times faster in some cases.

6.3 Backward-Facing Step Flows

The second validation problem is flow past a backward-facing step. The main feature of the flow is the presence just downstream of the step of a recirculation zone whose length increases with the Reynolds number. The low Reynolds number flow over a backward-facing step (the sudden expansion) is computed using the Cartesian cell based approach. The results are compared to the original PAMG code results at Reynolds number of 100 and 500. For both Reynolds number conditions, the geometry of the calculated domain is the same as indicated in Figure 6.11. The density ρ of the fluid is 1. The size of the calculated domain is 15.0 long and 1.0 high, the position of the step is located at x = 3.0 and the height of the step h is 0.5. The coarsest grid contains 576 cells and its resolution is $\Delta x = 0.1875$ and $\Delta y = 0.125$. It is important that the computational domain is long compared with the height of the step, otherwise the boundary conditions interfere with the recirculation zone.

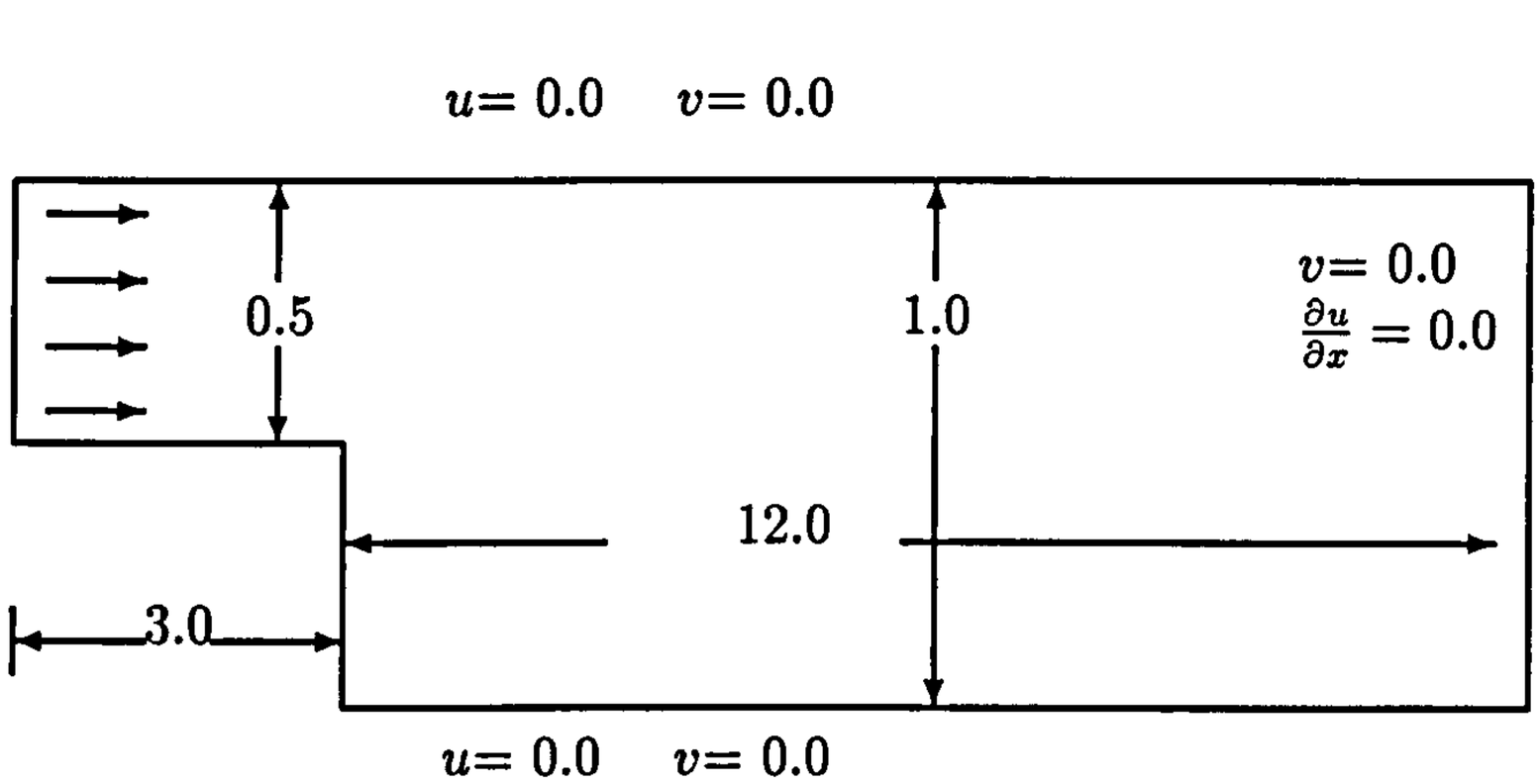


Figure 6.11: The Backward Facing Steps Schematic

As usual, we have specified a parabolic inlet profile (assuming fully developed flow), and we have full developed outlet conditions. The inflow velocity profile is fully developed:

$$u(y) = 16.0y(0.5 - y), \quad v(0, y) = 0.0 \tag{6.50}$$

No-slip boundary condition are applied on the all walls. In this configuration, we calculate the flow upstream of the step, thereby solving for the flow around the step and not ignoring the singularity at the corner. The important features of this flow are the recirculation zone behind the step and the velocity profiles downstream of the step which we discuss later in this section

More details of the investigation of this problem with the PAMG code can be seen in Thompson et al.[117]. It was thoroughly tested for both the uniform grid and adapted grid calculations. They demonstrated the effectiveness and accuracy of the adaptive multigrid techniques by considering the backward facing step problem and others well-known cases. P. Lezeau [63] also tested this particular case in his Ph. D. Thesis with a modified PAMG-multiphase code, comparing these two solvers, and found that the PAMG code and modified PAMG-multiphase code have the same accuracy and convergence factor. He also compared his results with the commercial CFD code, CFX 4.1. They have nearly the same results but the adaptive PAMG code has a quick convergence rate. Only the performance of the PAMG code and the newly modified PAMG code which introduced the Cartesian cut-cell to deal with the complex geometry conditions have been investigated in this section.

6.3.1 Reynolds Number 100

A coarse base grid is generated and both schemes converge through the requested seven levels of adaptive mesh refinement. Figure 6.12 shows some streamlines computed from the modified PAMG solution. The recirculation zone, which characterises this type of flow, appears clearly just downstream of the step. Note that the aspect ratio of the vertical and horizontal distances is not in scale on this figure. A close-up of the adapted grid near the backstep at the final level of mesh refinement is shown in Figure 6.13. Figure 6.14 shows the improvement of the solutions due to mesh refinement, at a given location in the backstep, where there is a significant reversed

flow region. Both schemes are compared to each other at a selected series of locations in Figure 6.15 to Figure 6.21.

As we expected, the modified PAMG solutions and the original PAMG results agree quite well along the vertical line x = 3.3 and horizontal lines (y = 0.25 in the middle of the inlet and y = -0.25 in the middle of the step). From these present solutions, the main conclusions we obtained are given below.

The fluids will be greatly decelerated because of the step since a great cross-section is available to them. Consequently the pressure gradient will become less steep downstream of the step. The change in the pressure gradients at the step is readily apparent. A small recirculation zone can be observed next to the step. It is small due to the smaller Reynolds number (equal 100) but appears very clearly on the solution profiles (see Figure 6.12). In Figure 6.14 and Figure 6.15 which show the horizontal velocity profile and vertical velocity profile along the line x= 3.3, it can be seen that the horizontal velocity is negative and the vertical velocity is positive at the bottom of the channel. All these features are in agreement with benchmark calculations and experimental observation and validate the new discretization [115].

The Performance of the two Solvers

We have established the accuracy of the solutions provided by the modified PAMG code. We will now focus on the efficiency of the solver and convergence factors observed with the Cartesian cut-cell algorithm. Since the exact error is not available in general, convergence is usually monitored by the reduction of the residual, measured using a suitable vector norm. The quantity which best qualifies efficiency is the convergence factor.

In this section we therefore concentrate on the convergence factors observed with the modified PAMG code. The first observation is that multigriding greatly improves the convergence rates of the single grid solver – FAS multigrid procedures are very

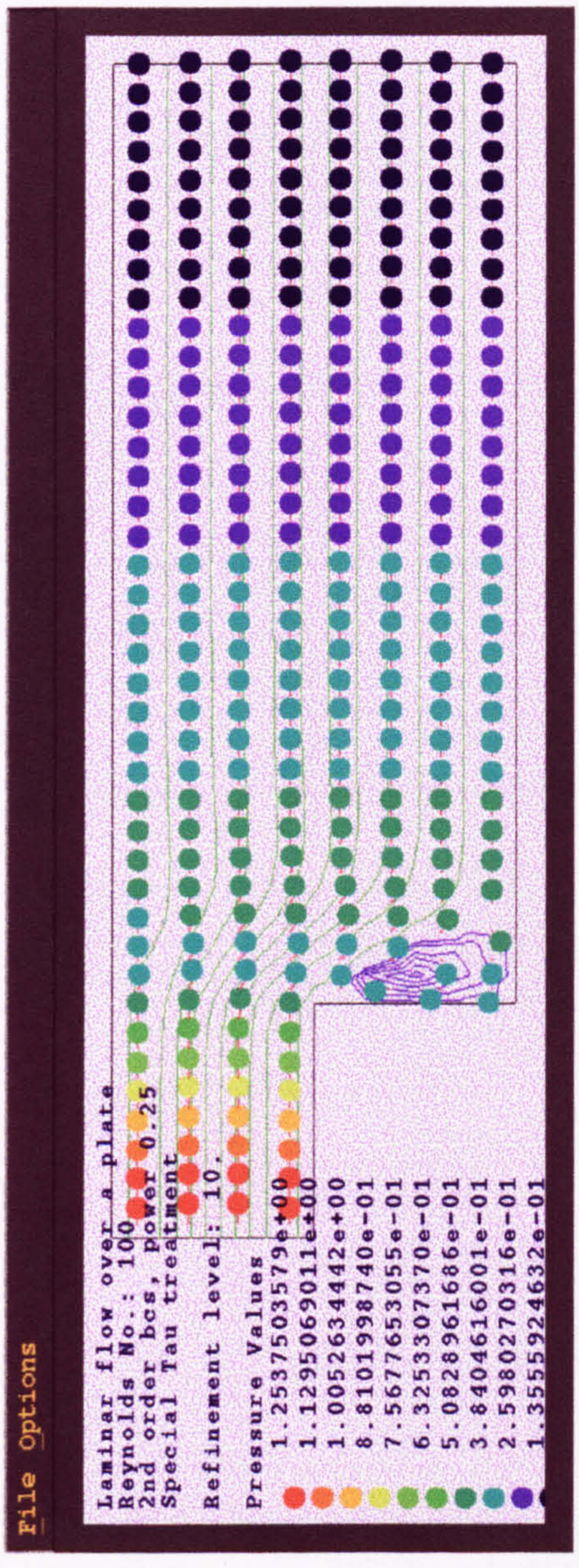


Figure 6.12: Backward Facing Step Problem - Streamlines and Pressure Distributions for the Modified PAMG Solution at Reynolds Number = 100

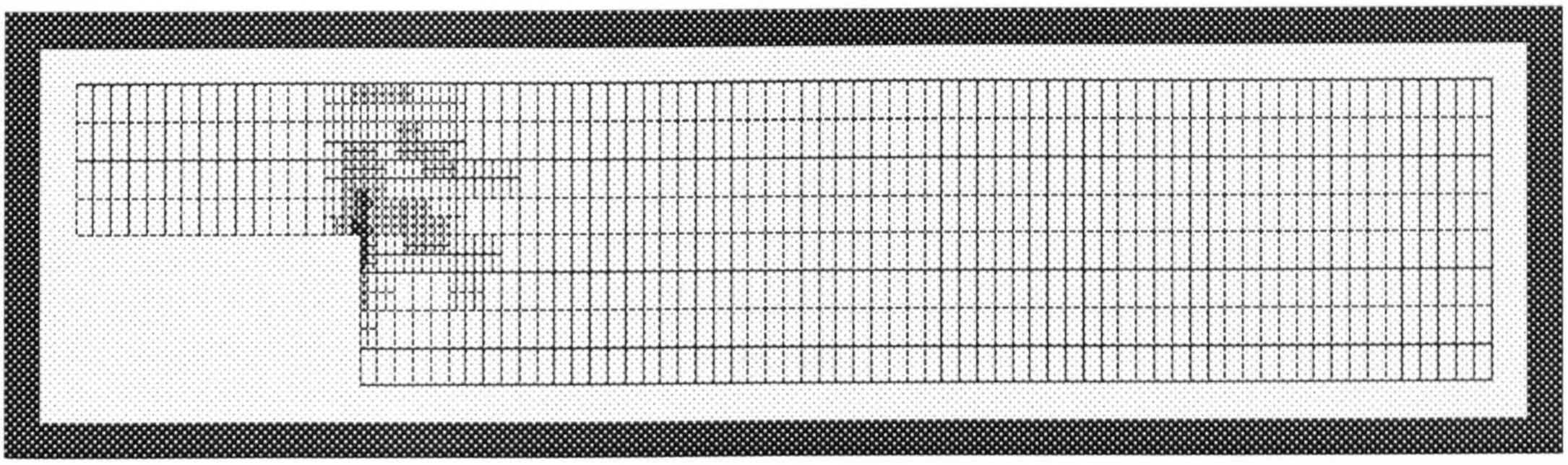


Figure 6.13: Backward Facing Step Problem - Final Adapted Grid on an Level 7 for the Modified PAMG Solution at Reynolds Number = 100

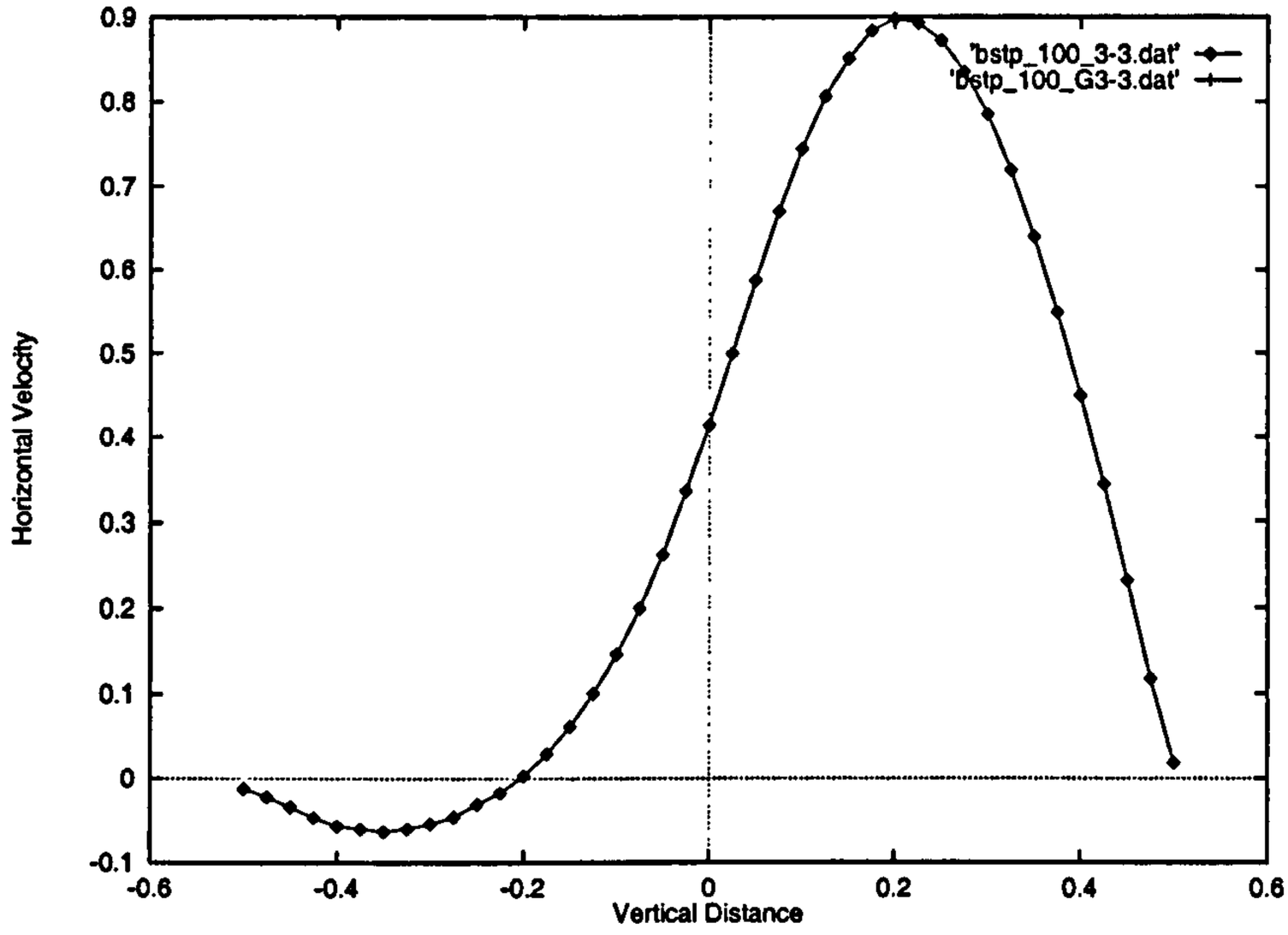


Figure 6.14: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line X = 3.3. Comparison of the PAMG and Modified PAMG solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

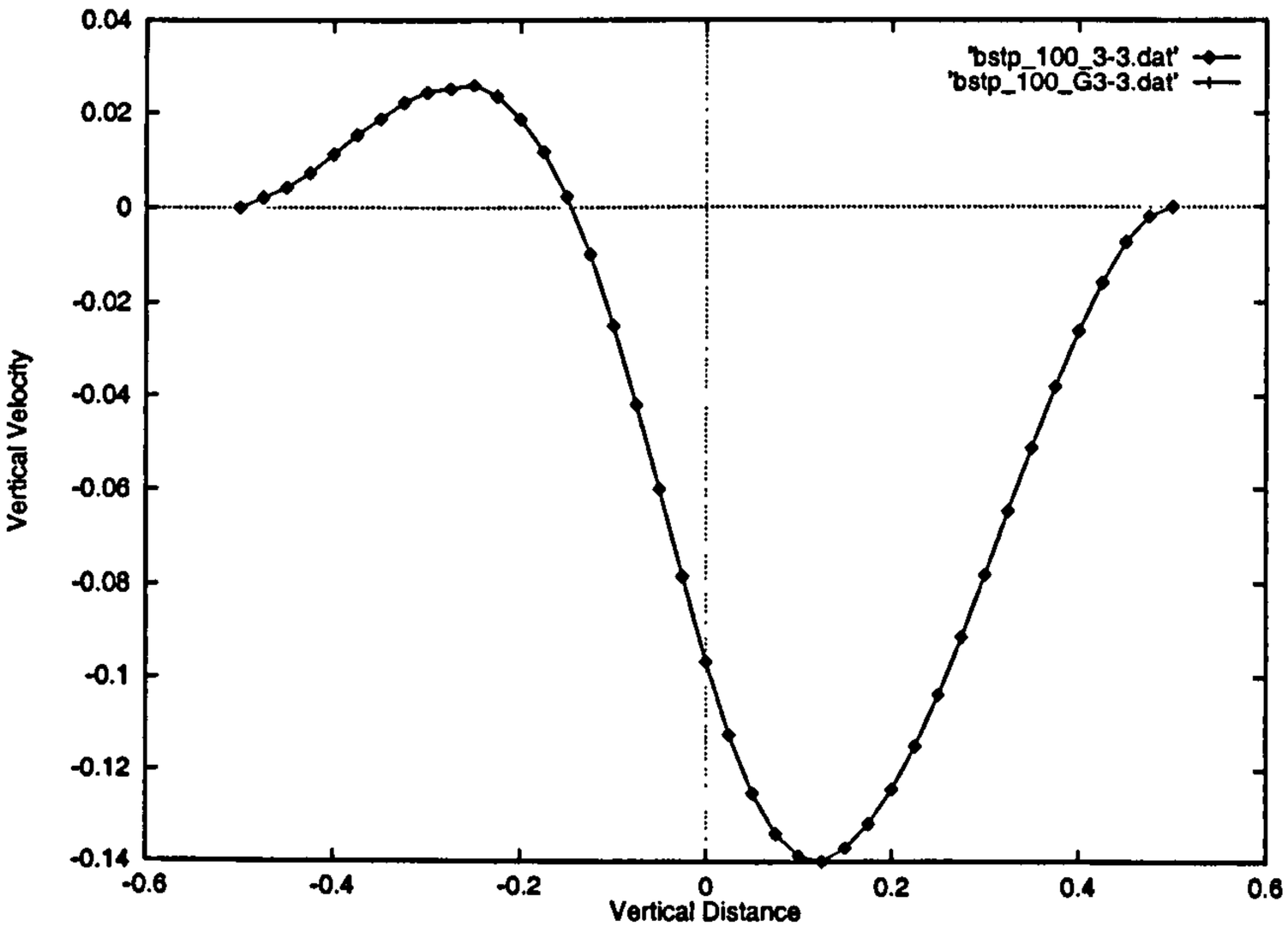


Figure 6.15: Backward Facing Step Problem - Vertical Velocity Profile Along the Line X = 3.3. Comparison of the PAMG and Modified PAMG solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

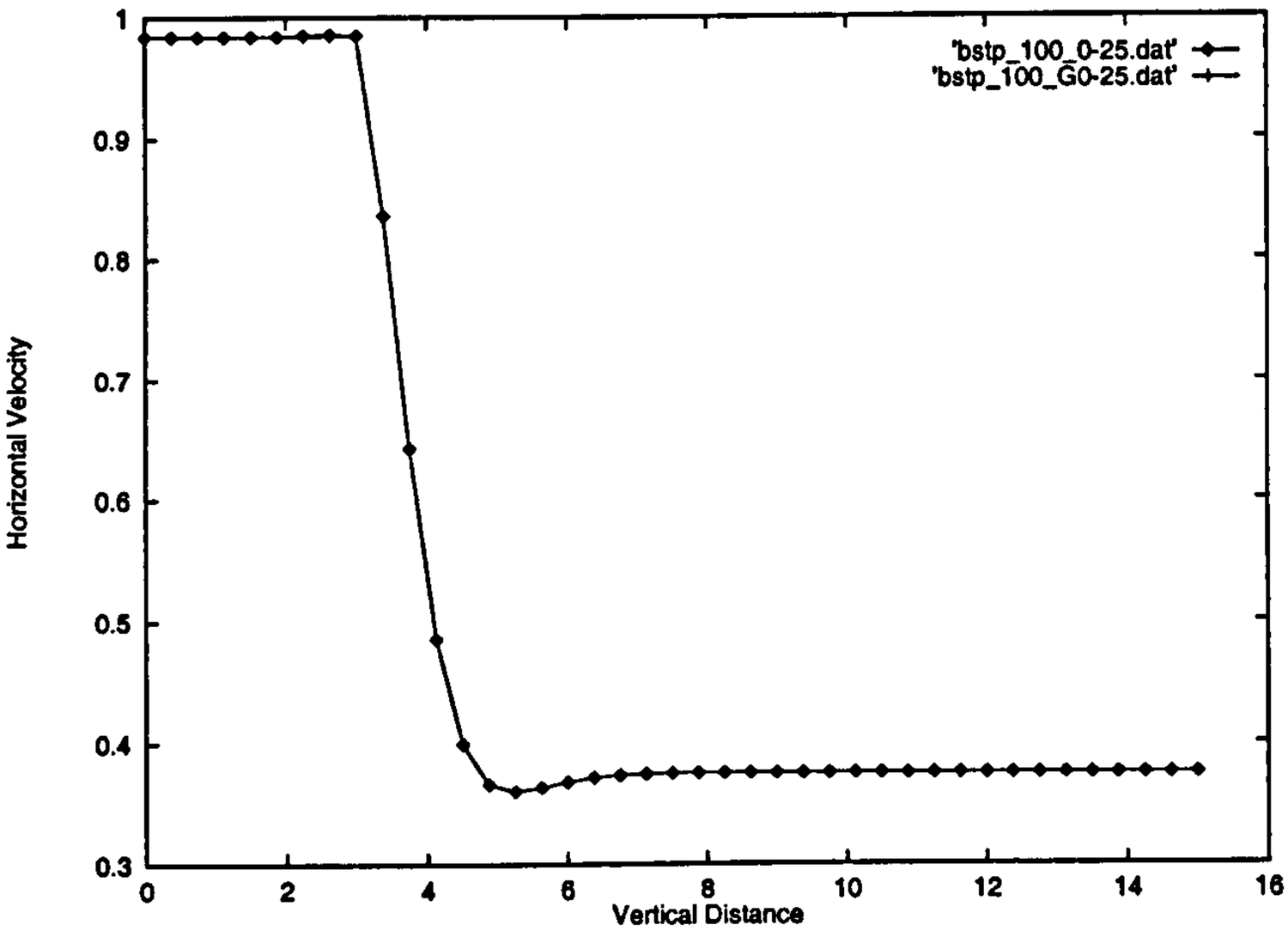


Figure 6.16: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line y = 0.25. Showing the Fluid Deceleration Through the Step in both algorithms.

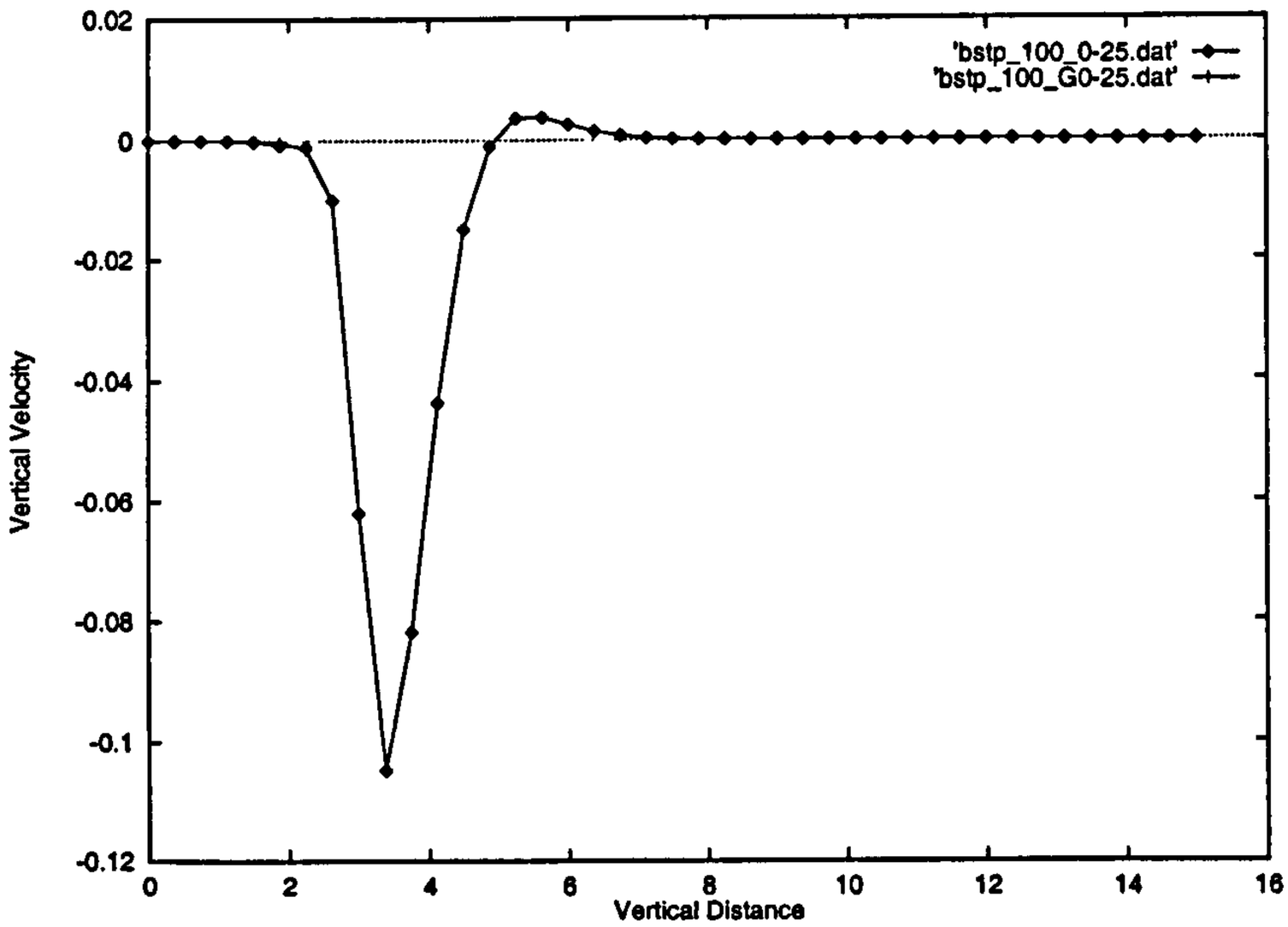


Figure 6.17: Backward Facing Step Problem - Vertical Velocity Profile Along the Line y = 0.25. Comparison of the PAMG and Modified PAMG solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

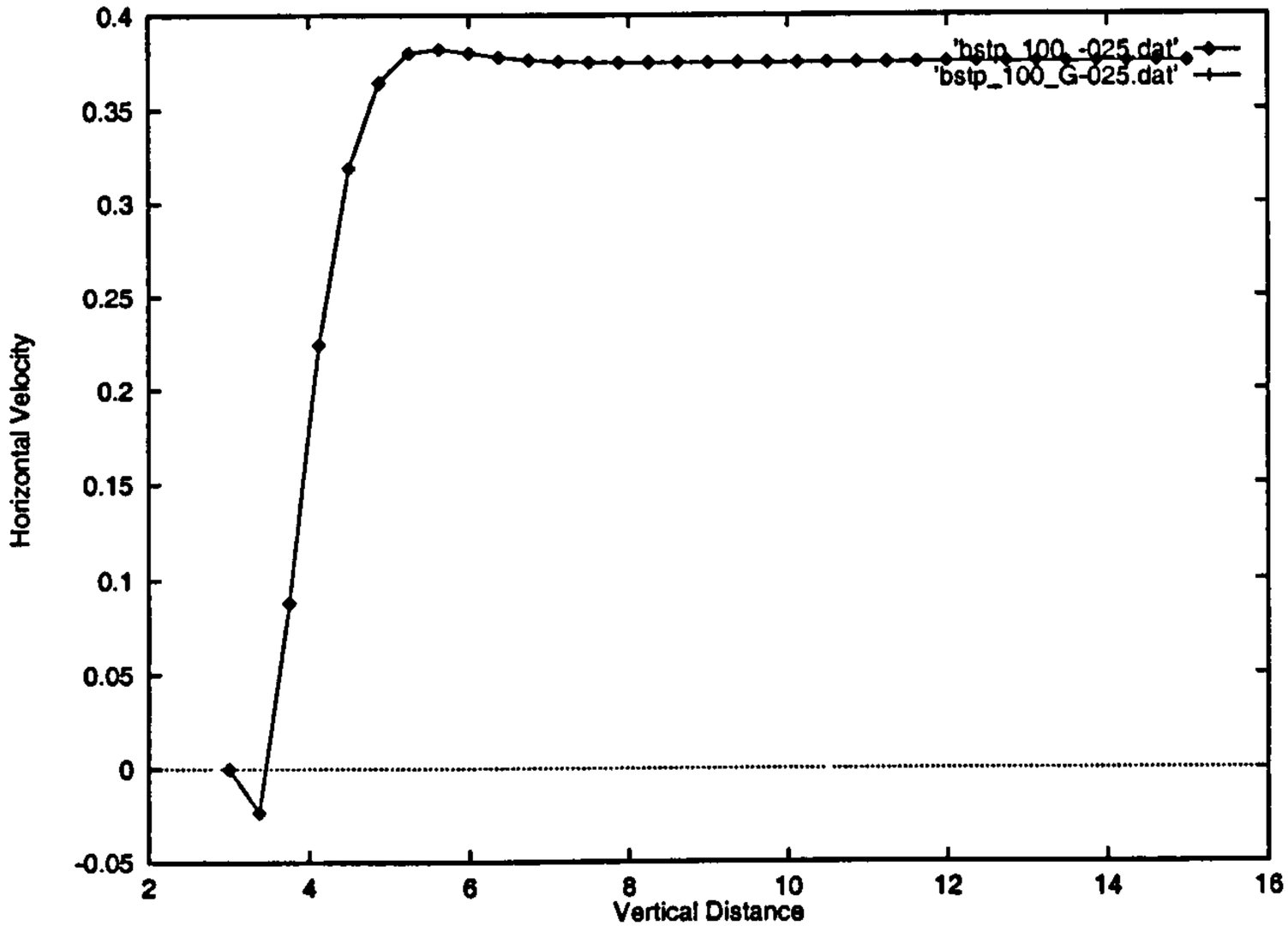


Figure 6.18: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line Y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

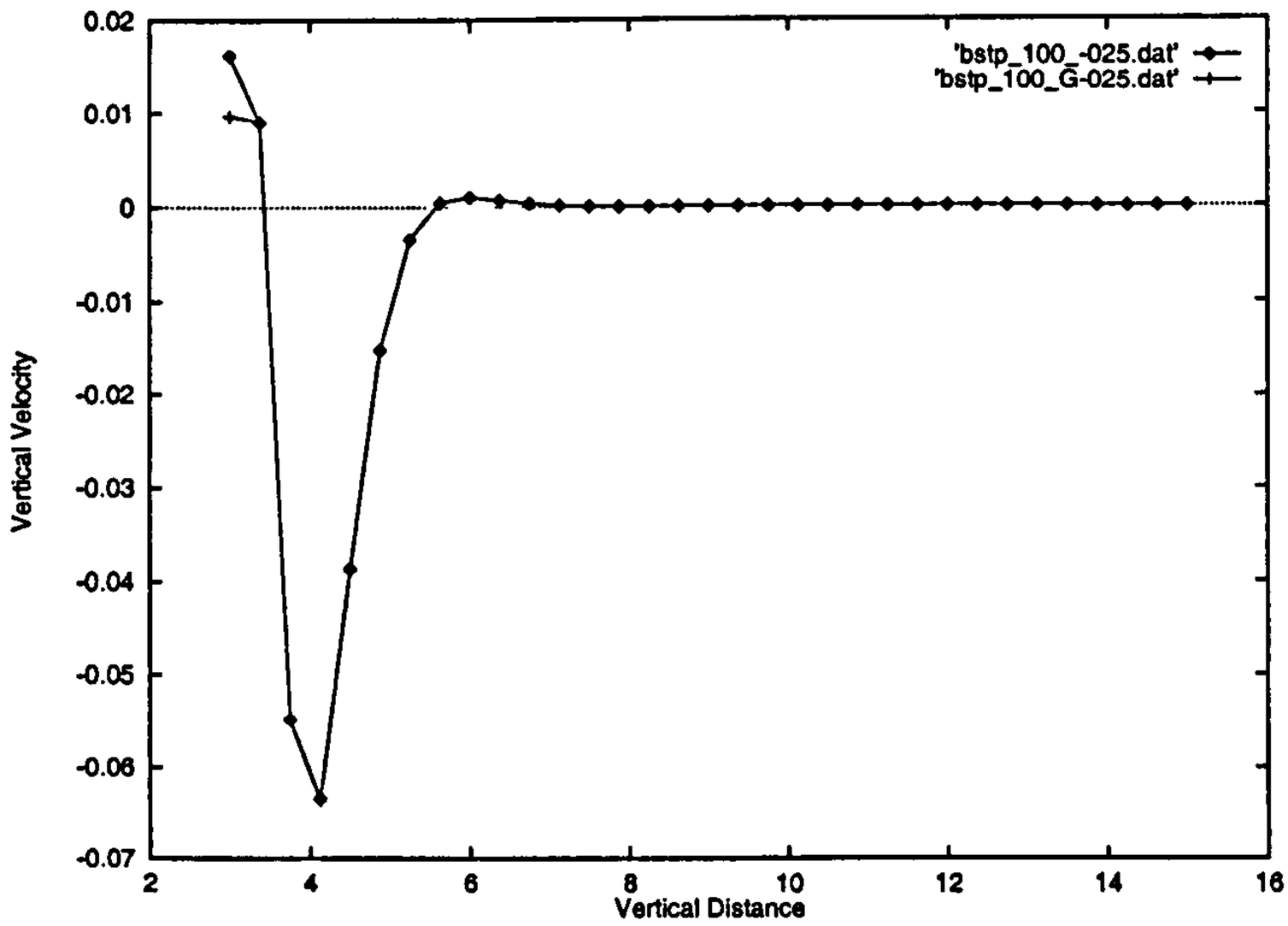


Figure 6.19: Backward Facing Step Problem - Vertical Velocity Profile Along the Line y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

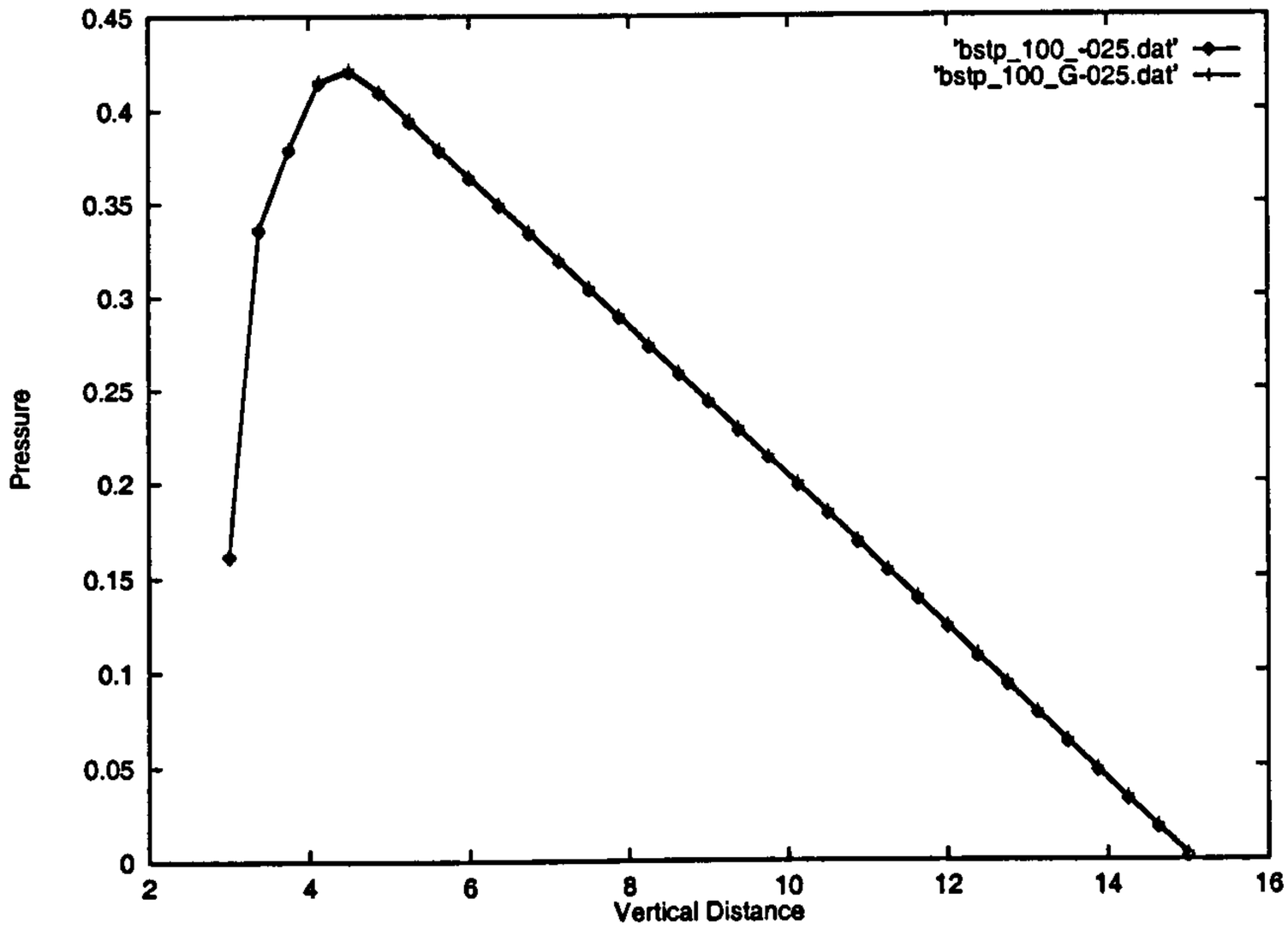


Figure 6.20: Backward Facing Step Problem - Pressure Profile Along the Line y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

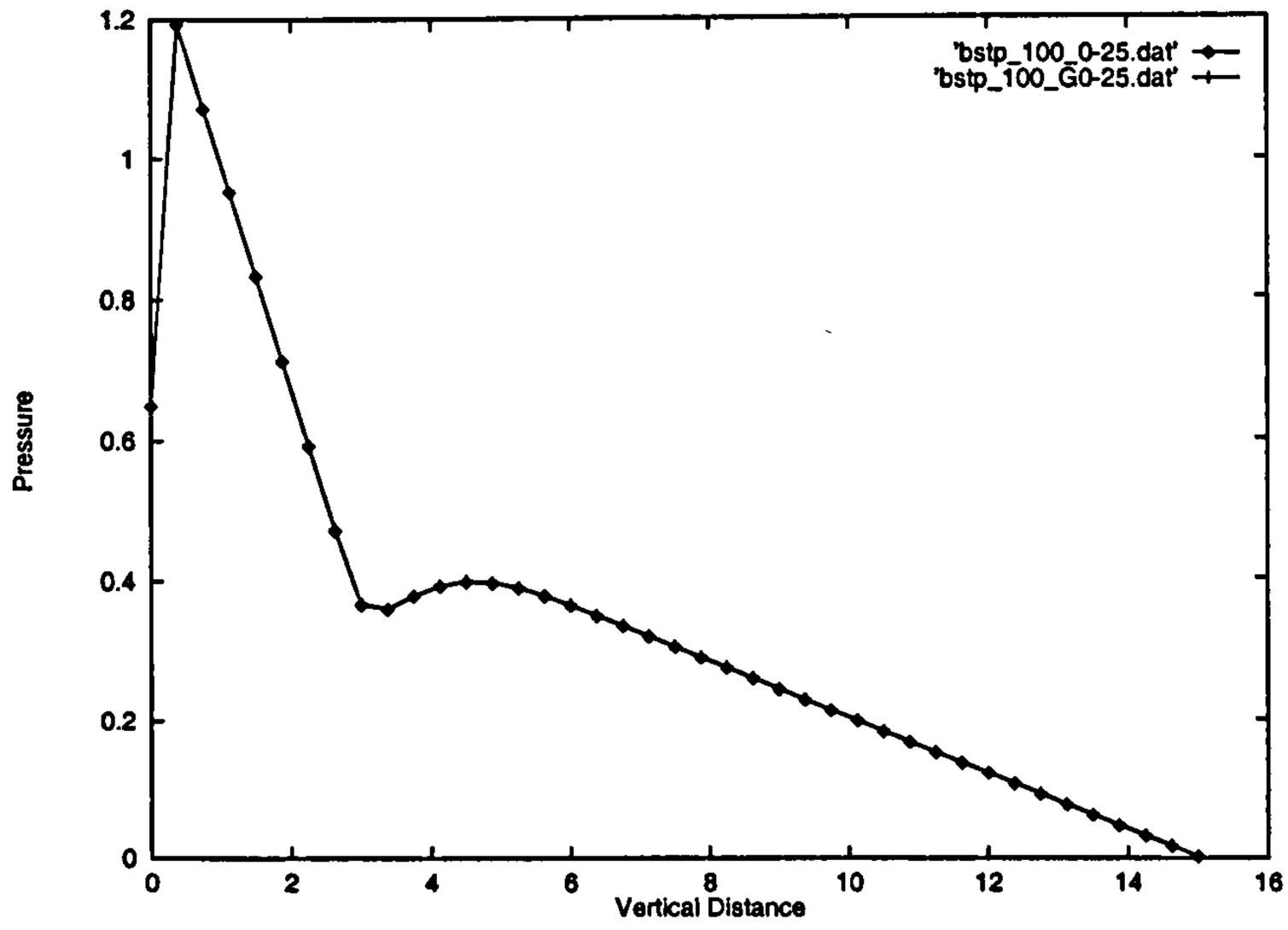


Figure 6.21: Backward Facing Step Problem - Pressure Profile Along the Line y = 0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7. Showing good agreement between the new and the standard algorithm.

successful acceleration techniques when applied to multigrid scheme. The ideal case would be that the convergence factors are grid-independent, ensuring that the method is optimal order-wise. It should be noted that convergence factors which to a large extent are grid independent, are observed in parts of the convergence histories. This indicates that the multigrid procedures can be expected to degrade the relative convergence factors caused by specific features of the equations.

For the backward facing step problem, a level 3 computation using an F-cycle is twice as fast as the corresponding level 2 computation (see Figure 6.22) and it was verified that the solutions are almost identical (Figure 6.23). Convergence histories for adapted multigrid computations from level 2 to level 7, when the original grid size is $\Delta x \times \Delta y$, are shown in Figure 6.22.

Figure 6.24 shows the convergence histories of the two schemes. Adaptive mesh refinement is made according to the convective criteria, and no attempts are made to modify the criteria. The mesh refinement improves the solution through each level of refinement, and both schemes perform well. However, the modified PAMG algorithm converged quicker than the original PAMG scheme. For example, at higher refinement grids (level 5, 6, 7), the original scheme needs around 12 cycles to reach the tolerance limit, but the modified PAMG scheme only needs 7 cycles to reach the tolerance. The improvement of the solution interpolation have made the convergence much quick at the non-cut-cell situation. It is very clear that the CC-PAMG algorithm has much smaller residuals after the first iteration cycles than the original PAMG. The reason is we use the multigrid FMG (see Figure A.4) algorithm and start calculation from the coarsest grid, then prolongate the solutions to the finer grid which introduce big residuals. The modified solution transformation method reduced the residuals, improving the velocity approximation from first-order to third-order accurate near wall and pressure corrections from first-order to second-order in the all calculation domain (see details in section 5.1).

As can be see from Table 6.5, the modified PAMG - Cartesian cut-cell method

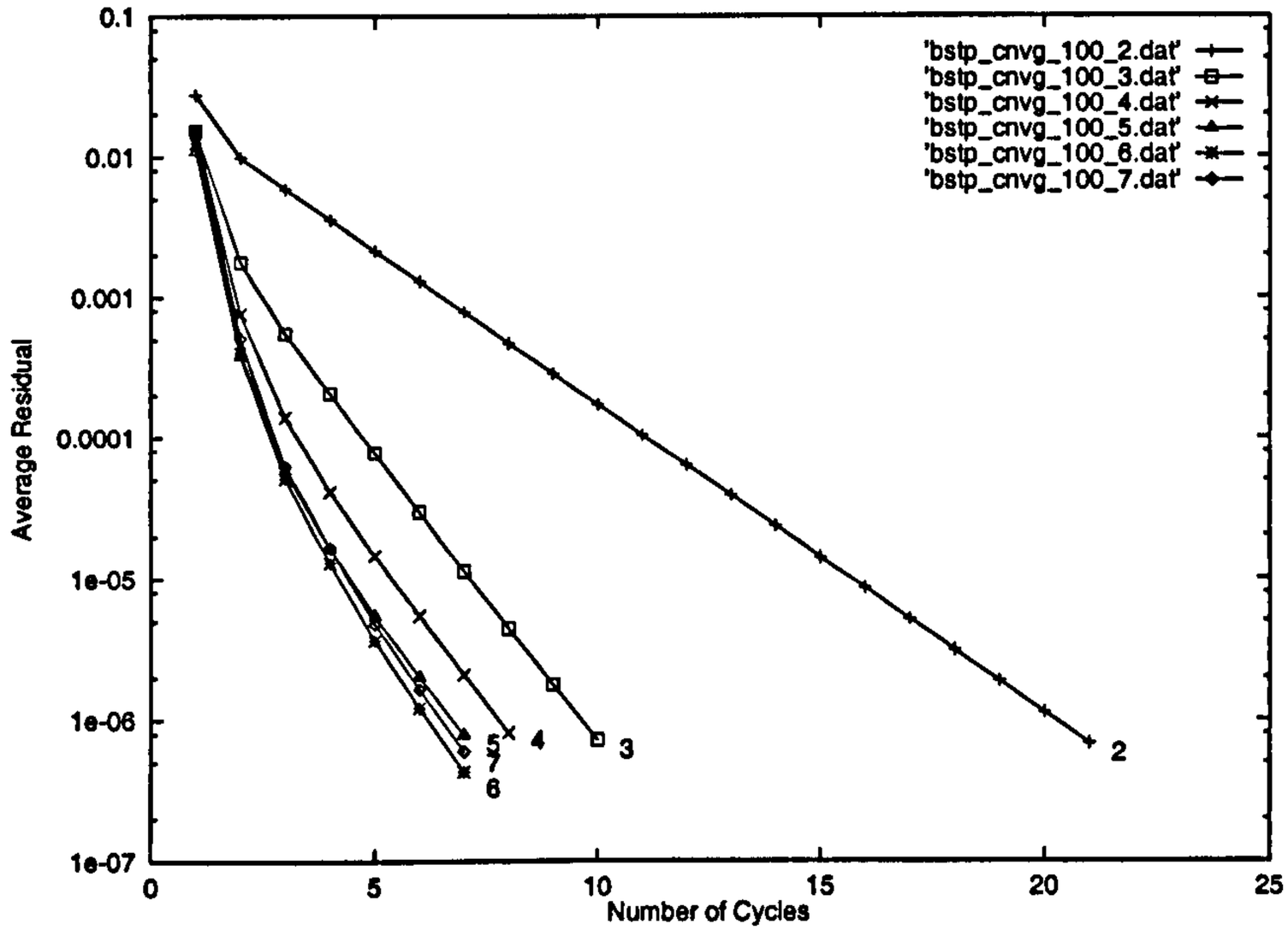


Figure 6.22: Backward Facing Step Problem - Convergence of Modified PAMG Solution for Adapted Multigrid Computation From Level 2 to Level 7

Figure 6.23: Backward Facing Step Problem - Horizontal Velocity Profiles Along the Line X = 3.3. Comparison of the Modified PAMG Solutions on Different Adaptive Grids

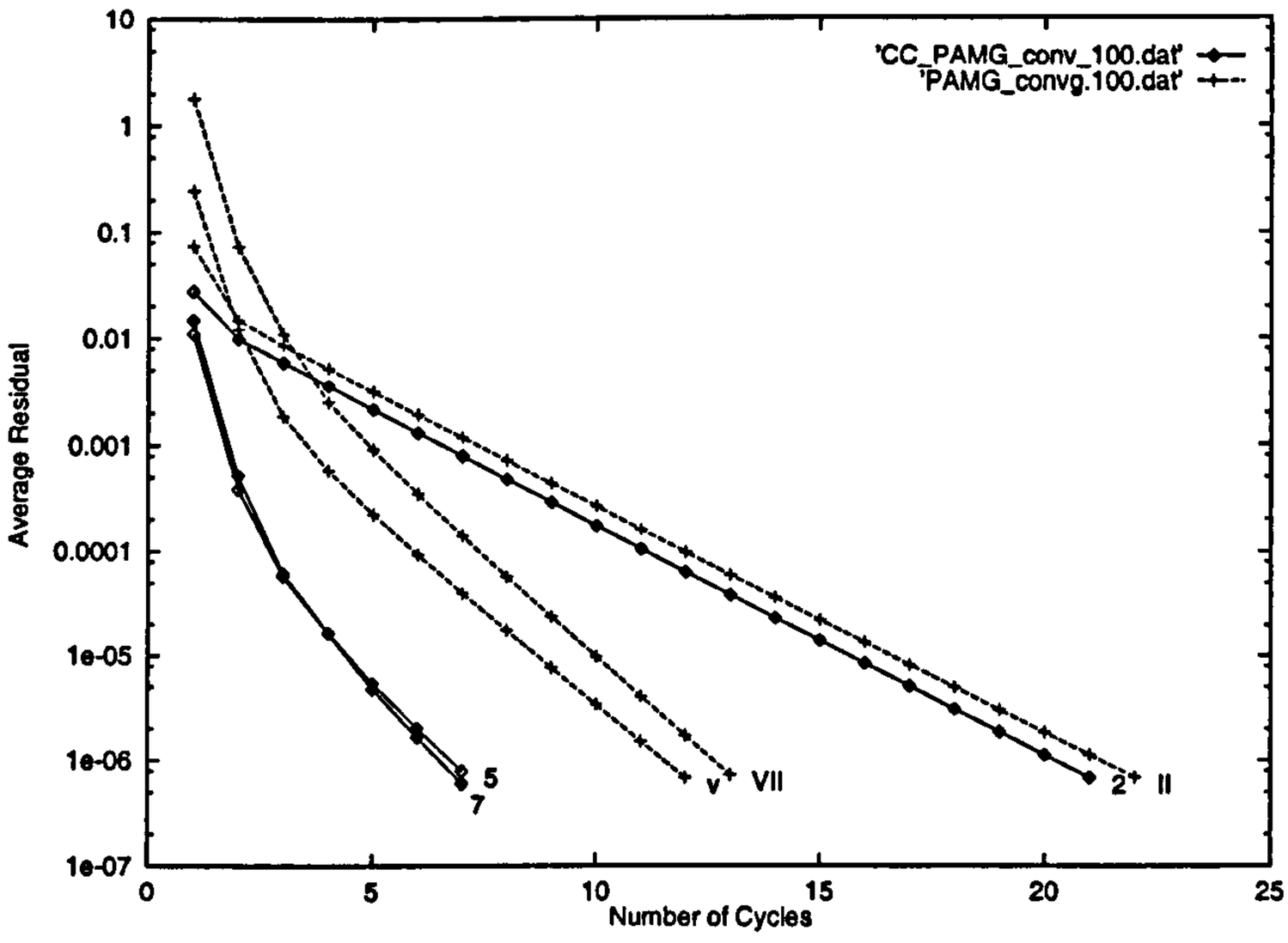


Figure 6.24: Backward Facing Step Problem - Convergence of Solution for Adapted Multigrid Computation. Comparison of the PAMG and CC-PAMG Solutions (indicated by numerical number) on an Adaptive Grid From Level 2, Level 5, and Level 7 which showing CC-PAMG is better.

scheme creates the same results as the original Adaptive Multigrid method, but rather more efficiently than the original multigrid method. In Table 6.5, we give the convergence rates and the total numbers of iterations for each patch[1] with the original PAMG and modified PAMG codes with Cartesian cut-cell algorithm. Definite improvement of the convergence rates is observed in the modified PAMG computations with Cartesian cut-cell method despite the complexity of the problem domain. However, the modified PAMG code with cut-cell approaching at higher level converge more quickly because the diffusion terms become more important than the convective terms as the mesh-size decreases. This effect is not see on the original PAMG calculations because the boundary condition treatment is improved. Comparing the numbers of relax iteration at different levels, the more efficient in the CC-PAMG is discovered. At level 2, this two algorithms have nearly the same (4266:4072) iterations to reach the tolerance. However, the PAMG code need more iteration ($\frac{1}{8} \sim \frac{1}{6}$ more iteration) than CC-PAMG algorithm at the higher levels (level 5,6,7). The convergence improves with PAMG but it is better with CC-PAMG.

6.3.2 Reynolds Number 500

For the higher Reynolds problem, the same coarse base grid is generated and after which adaptive mesh refinement is performed. Adaptive mesh refinement is desired for seven levels of refinement beyond the base grid as at Reynolds of 100. Figure 6.37 shows the convergence histories of the two schemes. Adaptive mesh refinement is made according to the convective criteria presented, and no attempts are made to modify the criteria. The mesh refinement improves the solution through each level of refinement, and both schemes perform well and yield nearly identical results. The recirculation zone, which clearly appears just downstream of the step, is much bigger

[1]In PAMG and CC-PAMG, the calculation take place by patches, Each patch has 4 × 4 cells or grids. It also includes guard place in each edge for the patch assembly. So the patch has 6 × 6 cells or grids. More detail can be see in section 5.2 where the cut-cell gives the whole patches which has been cut by half, and the non-cut-cells give the boundary between the inside cell and the guard place.

	Original PAMG				Modified PAMG			
Grid Level	Total Pats.	No. Iterat.	No. Cycles	Average Converg.	Total Pats.	No. Iterat.	No. Cycles	Average Converg.
2	180	4266	22	0.565	180	4072	21	0.542
3	223	4676	12	0.351	223	3896	10	0.342
4	293	5651	10	0.311	293	4714	8	0.289
5	318	6814	12	0.359	326	5550	7	0.251
6	336	8632	12	0.319	350	7004	7	0.223
7	353	9657	13	0.317	379	8448	7	0.228

Notes: F(2,2) cycles with Reynolds No.= 100, Power = 2.0, tolerance = 10^{-6}

Table 6.5: Rate of Convergence for the Backward Facing Step Problem: Original PAMG Calculations and Modified PAMG Calculations at Reynolds number =100

than the one at Reynolds number of 100 as expected (compare Figure 6.12 and 6.25).

Figure 6.27 shows the improvement of the solution due to mesh refinement for the two schemes, at a given location in the backstep, where there is a significant reversed flow region. The adapted grid near the backstep at the final level of mesh refinement for the modified PAMG scheme is shown in Figure 6.26. As in the previous case, direct comparisons between the two schemes are shown in Figure 6.27 to 6.34 at the seventh refinement level. As before, both schemes obtain profiles that are of comparable accuracy, so that it is difficult to see any appreciable differences in the velocity plots. Figure 6.36 illustrates the improvement of vertical velocity profiles in the solution quality automatically obtained with the adaptive mesh refinement.

The Performance of the two Solvers at Reynolds Number 500

There is little difference between the computed profiles from the two algorithms, What is very noticeable, however, is that the CC-PAMG scheme converges much quicker than the original PAMG algorithm on the higher levels. An examination of the rate

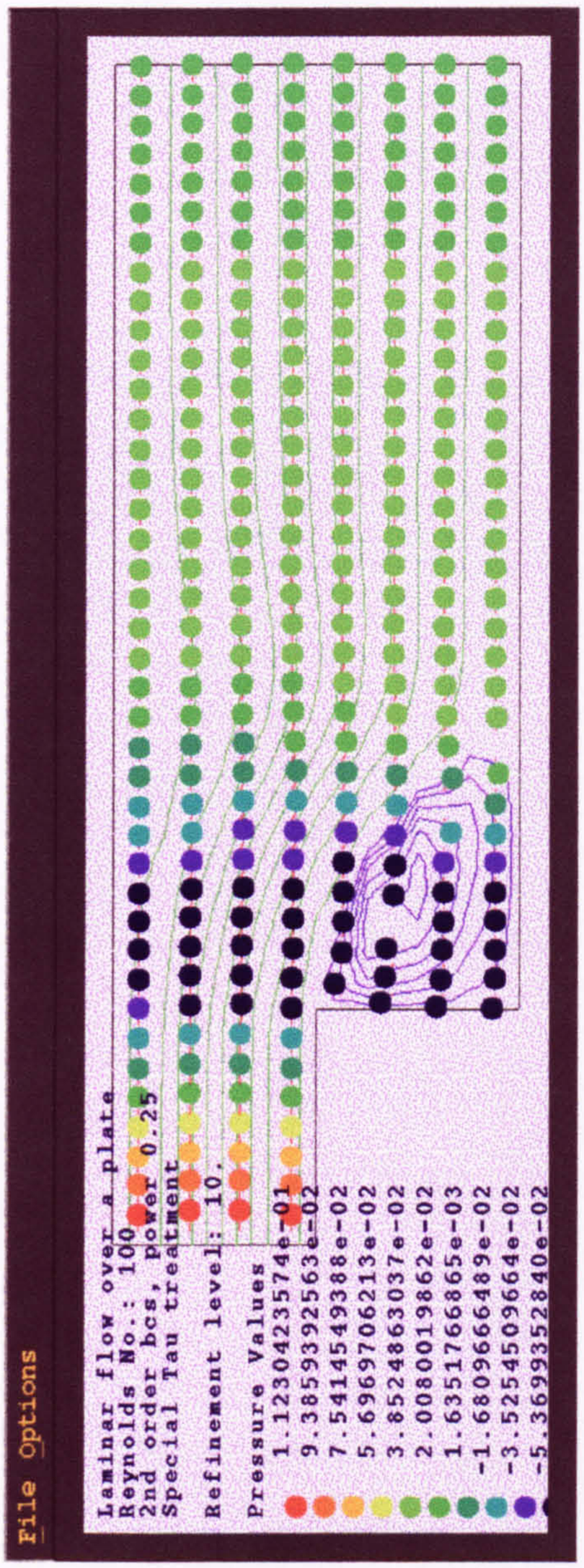


Figure 6.25: Backward Facing Step Problem - Streamlines and Pressure Distributions for the Modified PAMG Solution at Reynolds Number = 500

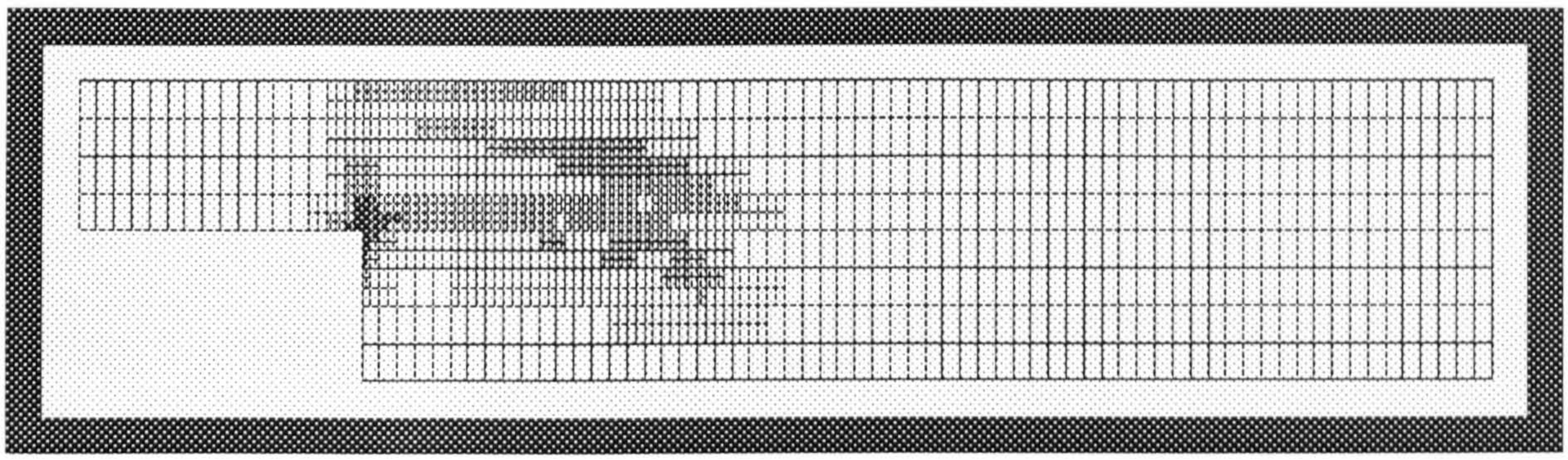


Figure 6.26: Backward Facing Step Problem - Final Adapted Grid on an Level 7 for the Modified PAMG Solution at Reynolds Number = 500

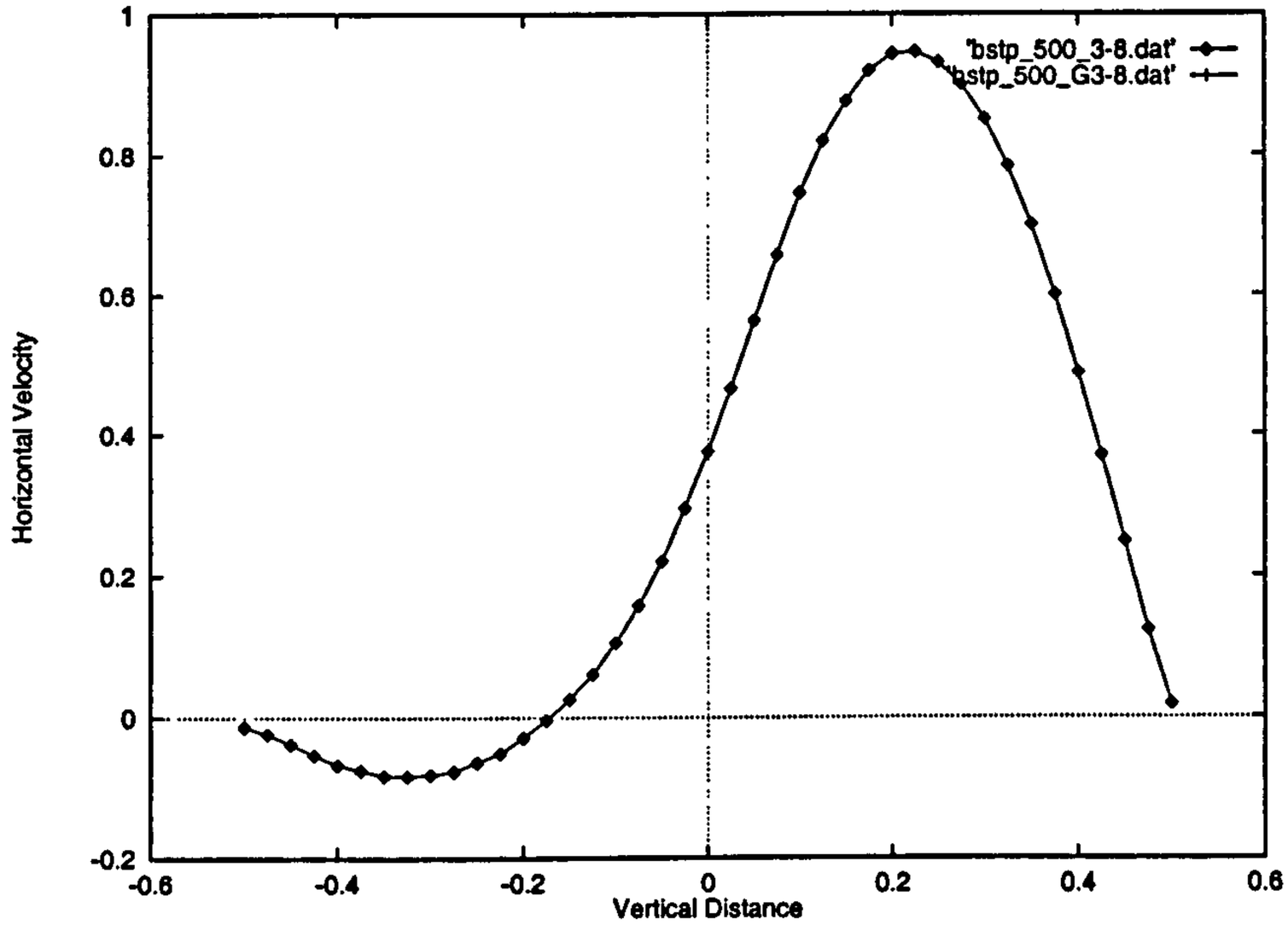


Figure 6.27: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line X = 3.8. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

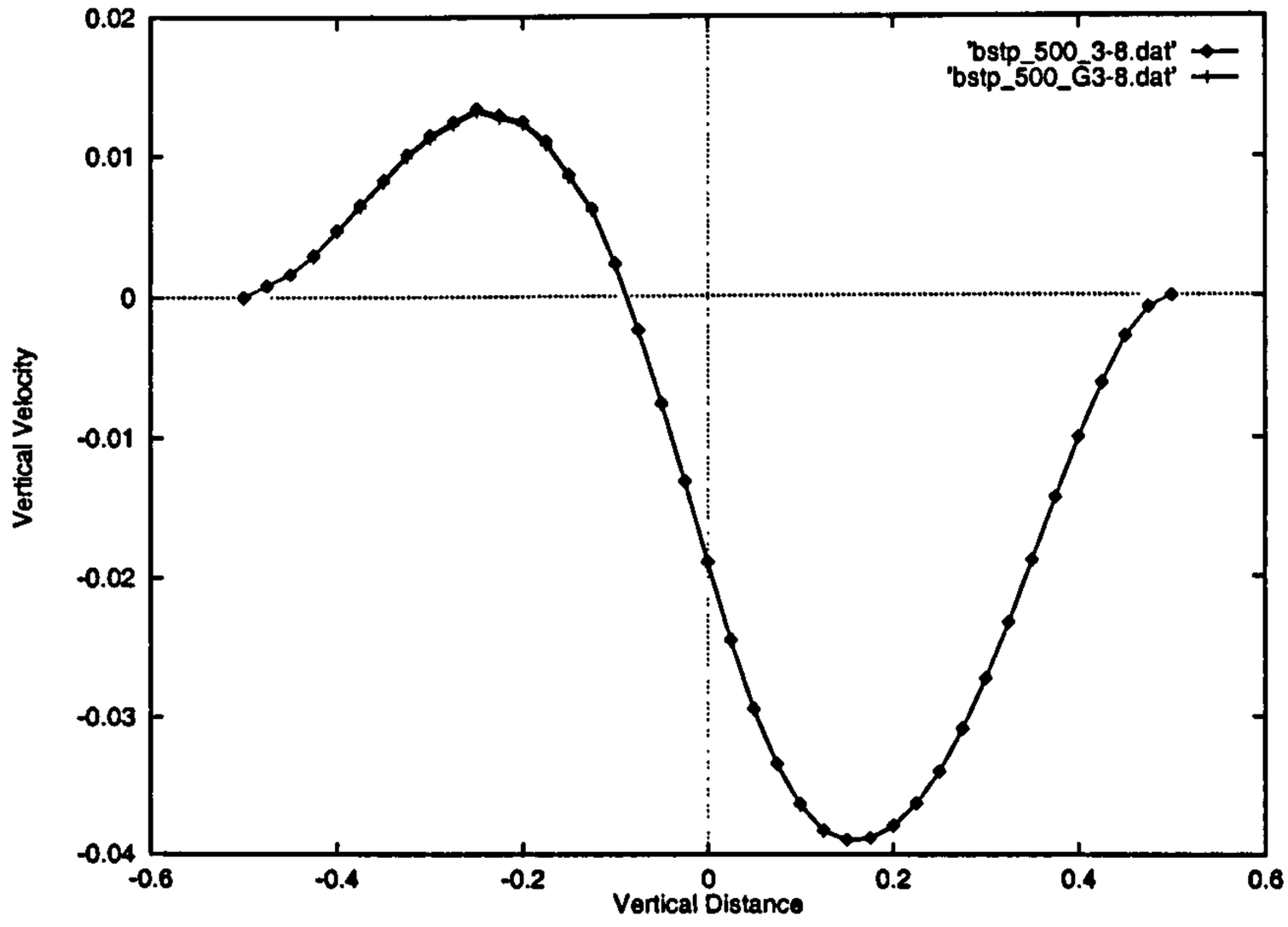


Figure 6.28: Backward Facing Step Problem - Vertical Velocity Profile Along the Line X = 3.8. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

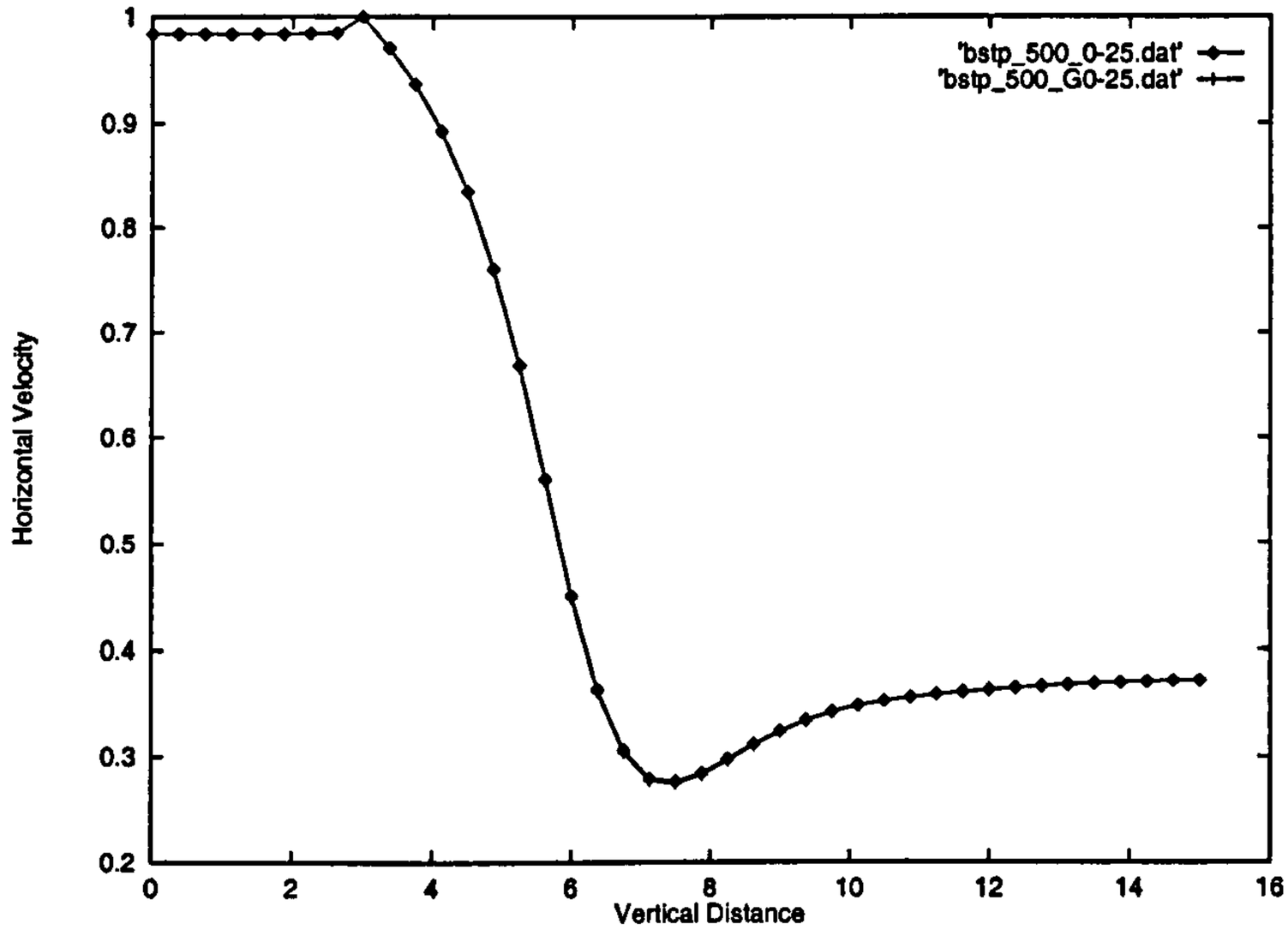


Figure 6.29: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line y = 0.25. Showing the Fluid Deceleration Through the Step,, Reynolds Number = 500

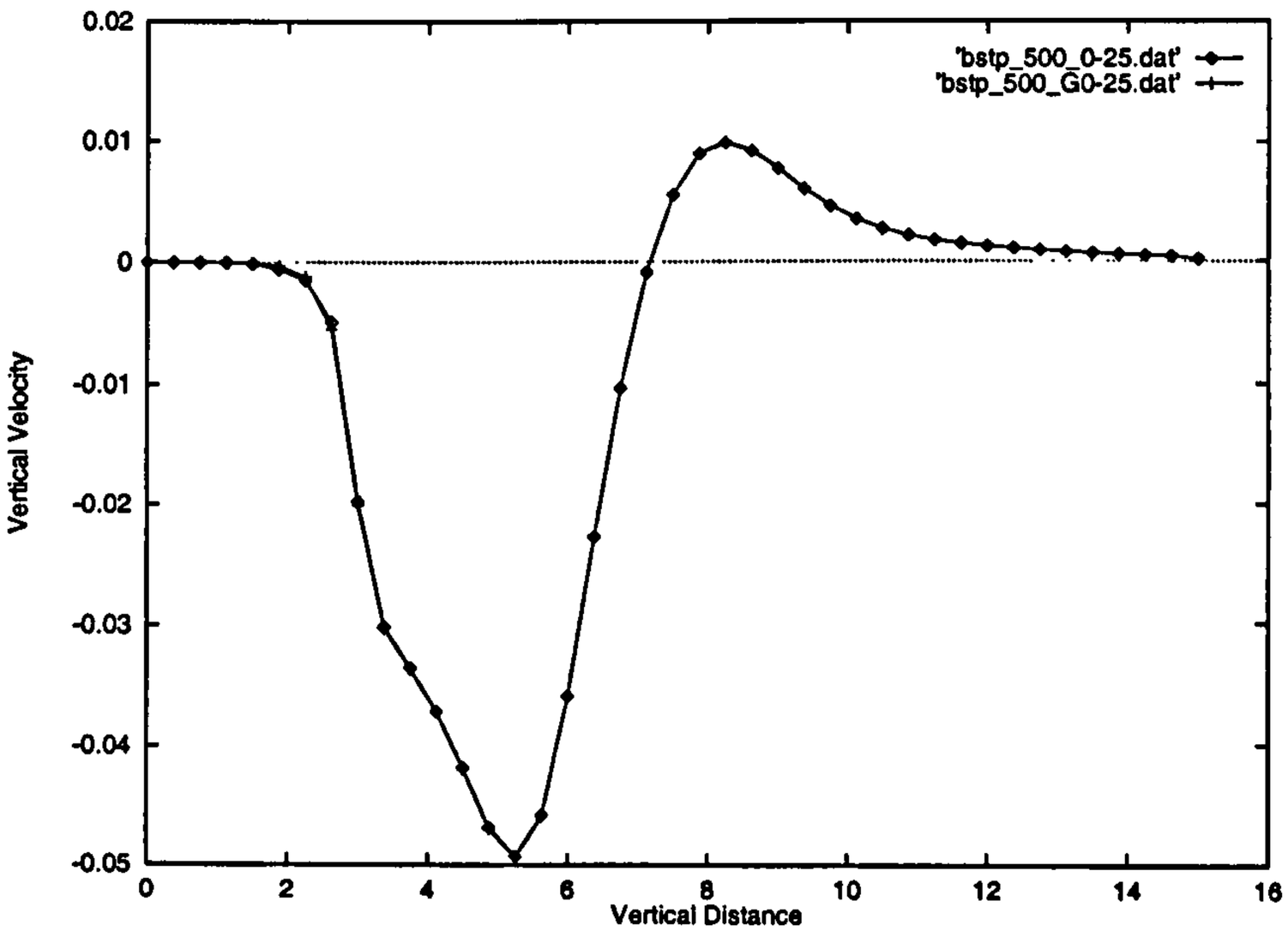


Figure 6.30: Backward Facing Step Problem - Vertical Velocity Profile Along the Line y = 0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

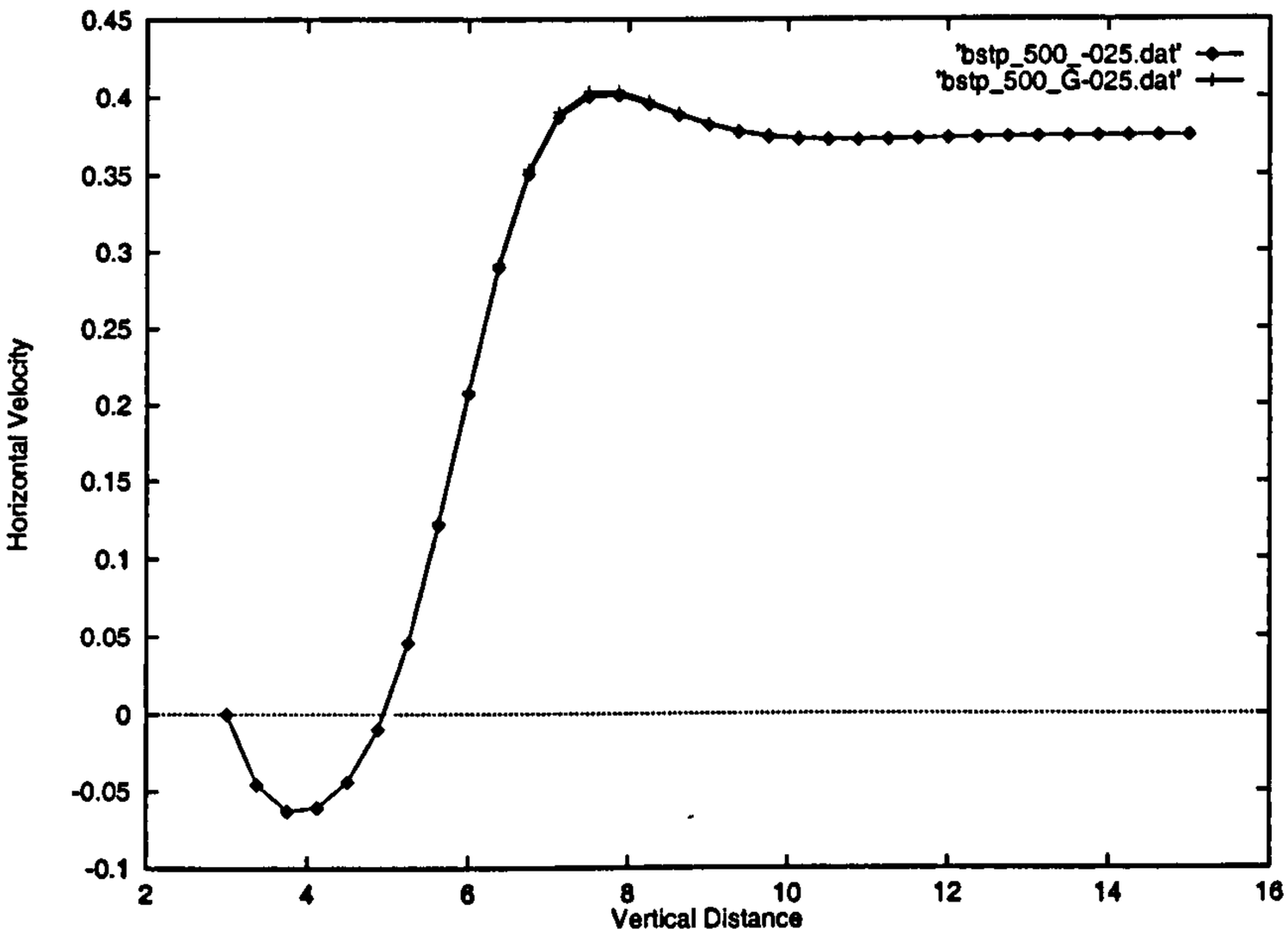


Figure 6.31: Backward Facing Step Problem - Horizontal Velocity Profile Along the Line Y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

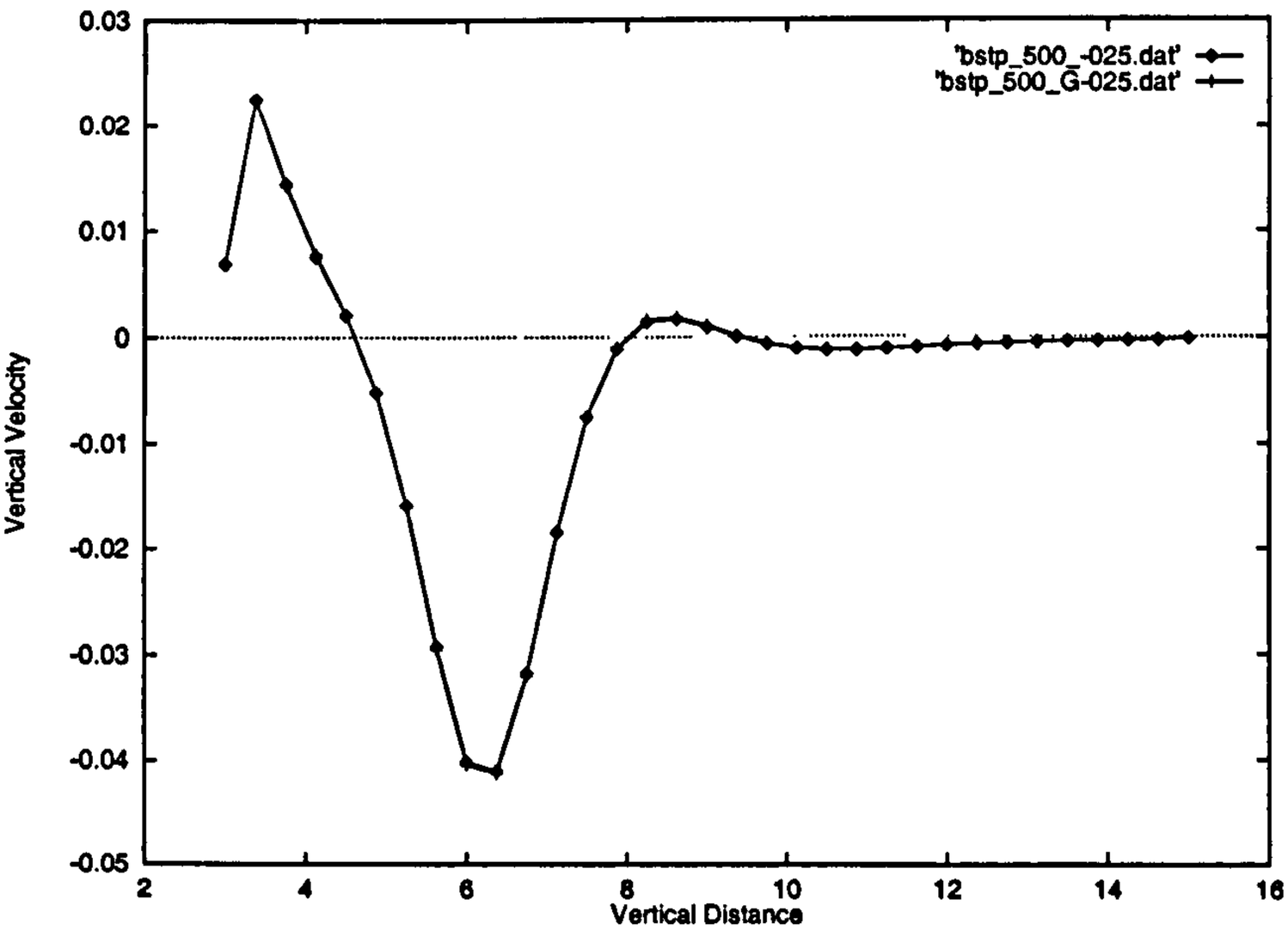


Figure 6.32: Backward Facing Step Problem - Vertical Velocity Profile Along the Line y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

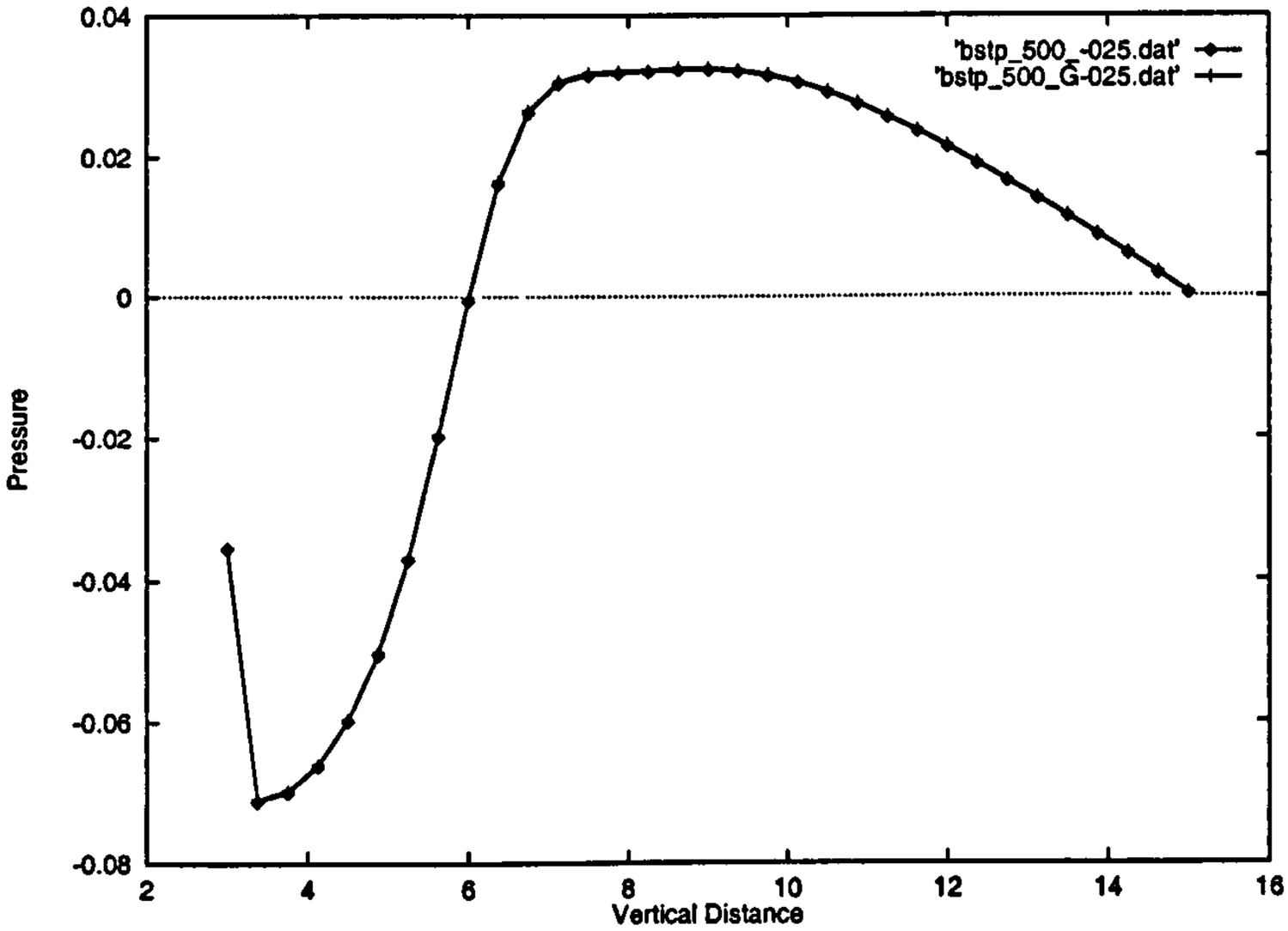


Figure 6.33: Backward Facing Step Problem - Pressure Profile Along the Line y = -0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

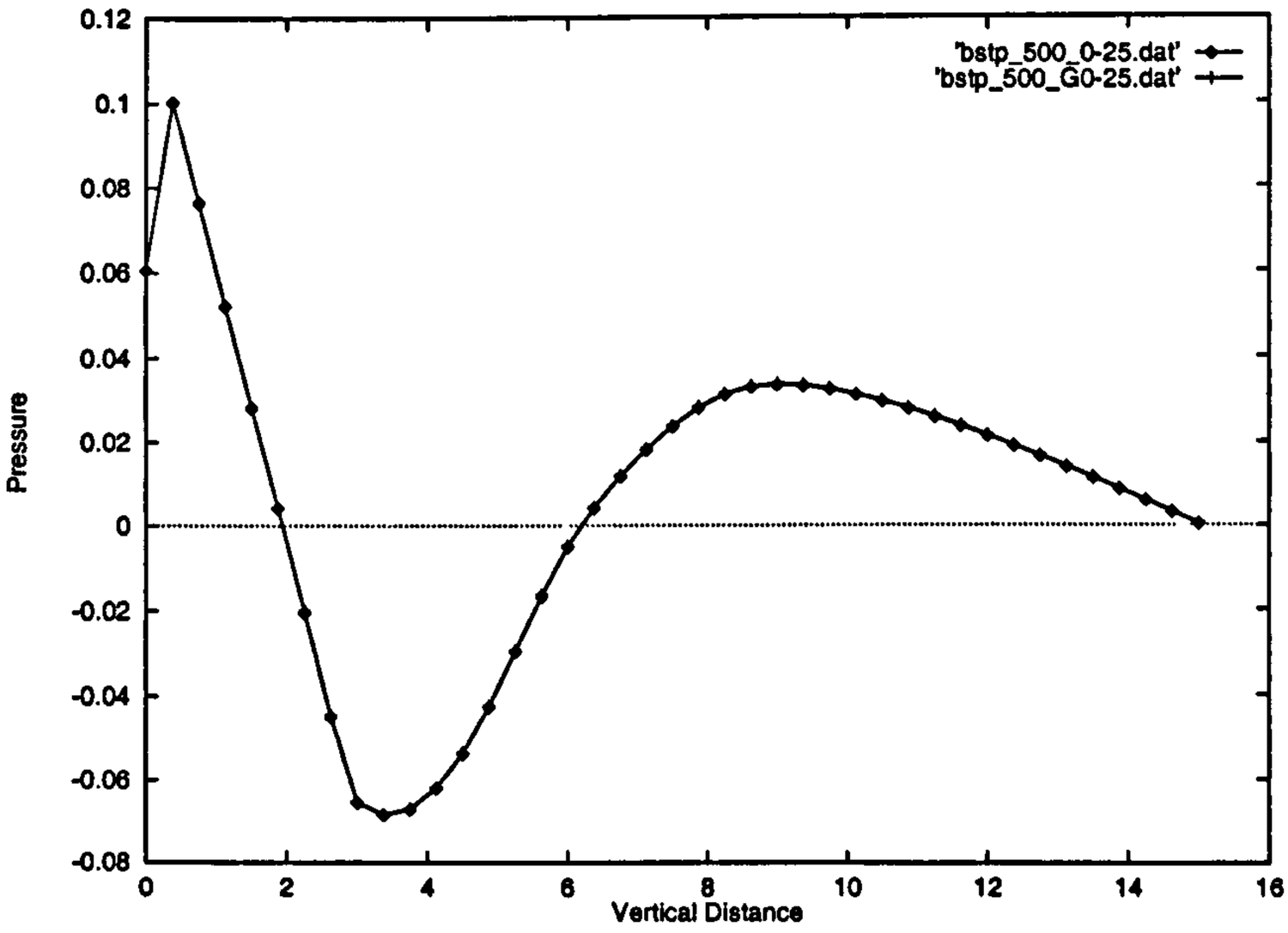


Figure 6.34: Backward Facing Step Problem - Pressure Profile Along the Line y = 0.25. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid Up to Level 7, Reynolds Number = 500

of convergence shows similar behaviour as in the lower Reynolds number case which exhibits in Figure 6.35. The modified PAMG scheme yielded an improved convergence ratio than the original scheme (see Figure 6.37).

As can be see from Table 6.6, the CC-PAMG - Cartesian cut-cell method scheme at Reynolds of 500 creates the same results as the original Adaptive Multigrid method, but is more efficient than the original multigrid method. At levels 2, 3, and 4, the two schemes at Reynolds of 500 have nearly the same convergence rate. however, at the higher levels (Level 5, 6, and 7), the CC-PAMG algorithm has the much quick reduction ratio than the original one. In Table 6.6, we give the convergence rates for the adapted grid calculation and the total numbers of relaxation for each patch with the original PAMG code and modified PAMG code with Cartesian cut-cell algorithm. Certain improvement of the convergence rates is observed in the modified PAMG computations with Cartesian cut-cell method despite the complexity of the problem domain. This effect is not see on the original PAMG calculations as in the lower Reynolds number of 100 due to the boundary condition treatment is improved. In level 5, about 28the same results as in CC-PAMG. Over one-third more relaxation should be added in level 6 for PAMG to reach the tolerance. There are two times iteration numbers for PAMG to have the same solution as CC-PAMG at level 7.

Conclusion

In this section, we have demonstrated the efficient and accuracy of our results by considering the well-known model problem — the backward-facing step flow.

We have shown that the rate of convergence we observed for the adaptive calculation has dramatically improved. The accuracy of our results is comparable to that of other benchmarks for both the uniform and the adaptive calculations. Additionally, we have shown that the interpolation that we use at interfaces is good and that we are easily able to accelerate the convergence rate. For the case we have considered, we have been able to achieve significant gains in speed compared to the original PAMG

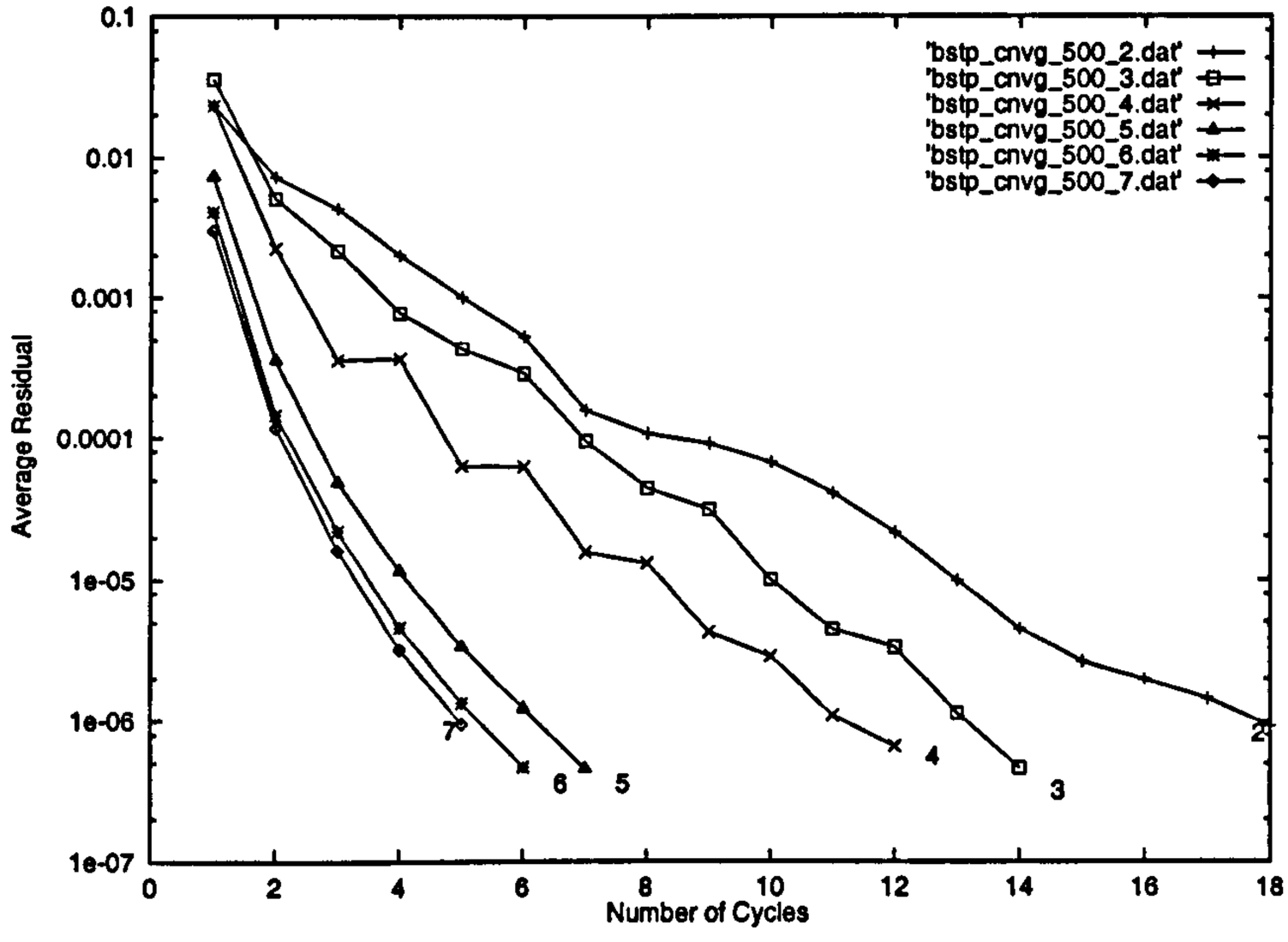


Figure 6.35: Backward Facing Step Problem - Convergence of Modified PAMG Solution for Adapted Multigrid Computation From Level 2 to Level 7, Reynolds Number = 500

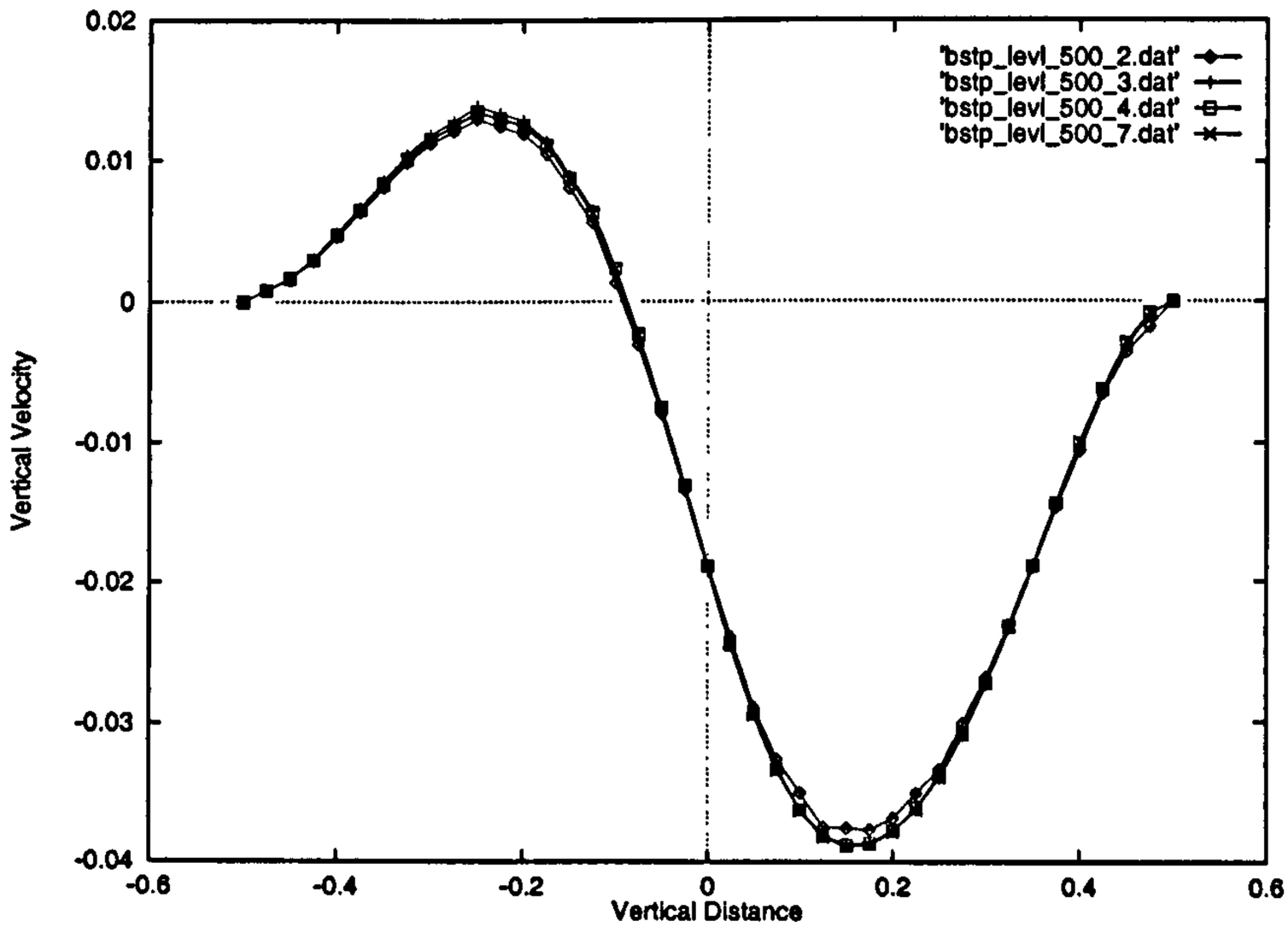


Figure 6.36: Backward Facing Step Problem - Vertical Velocity Profiles Along the Line X = 3.8. Comparison of the Modified PAMG Solutions on different Adaptive Grids, Reynolds Number = 500

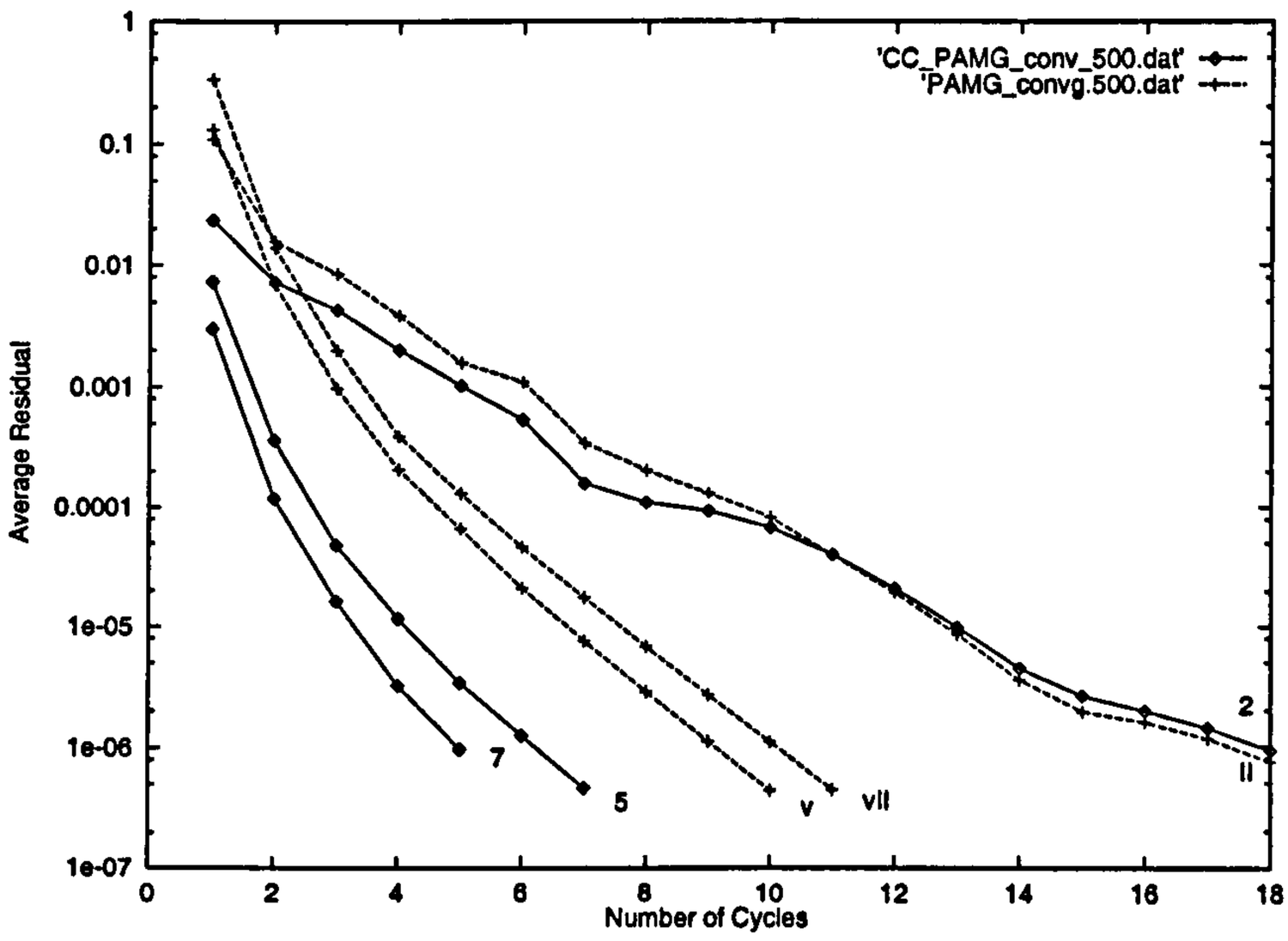


Figure 6.37: Backward Facing Step Problem - Convergence of Solution for Adapted Multigrid Computation. Comparison of the PAMG and Modified PAMG Solutions on an Adaptive Grid From Level 2, Level 5, and Level 7, Reynolds Number = 500

	Original PAMG				Modified PAMG			
Grid Level	Total Pats.	No. Iterat.	No. Cycles	Average Convergence	Total Pats.	No. iterat.	No. Cycles	Average Convergence
2	180	3482	18	0.528	180	3490	18	0.539
3	261	6642	18	0.502	261	5456	14	0.457
4	437	6523	12	0.355	435	5554	12	0.403
5	566	7138	10	0.361	553	5550	7	0.227
6	597	9298	10	0.377	589	6002	6	0.191
7	622	12803	11	0.358	624	6058	5	0.157

Notes: F(2,2) cycles with Reynolds No.= 500, Power = 2.0, tolerance = 10^{-6}

Table 6.6: Rate of Convergence for the Backward Facing Step Problem: Original PAMG Calculations and Modified PAMG Calculations at Reynolds number = 500

calculations.

6.4 Solution of Steady Uni-directional Channel Flow: – Non-aligned

The laminar channel flow is a well-known simple test case in flow dynamics. It has an analytical solution. For fully developed inlet flow, the theoretical velocity profiles are:

$$u = -\frac{1}{2\mu}\frac{dp}{dx}(by - y^2); \quad v = 0$$

which satisfies the continuity equation identically. Where b is the channel width. In order to expand the CC-PAMG code to deal with the complex geometry boundaries, a non-aligned calculation domain has been introduced. For this situation, the boundary conditions are no longer aligned with the base coordinate axes. The flow channel is rotated α degrees about the horizontal level (see Figure 6.38). For this problem, the velocity vector has the same direction everywhere, and is independent of distance in the flow direction. We can use the Navier-Stokes equations to solve this kind of flow.

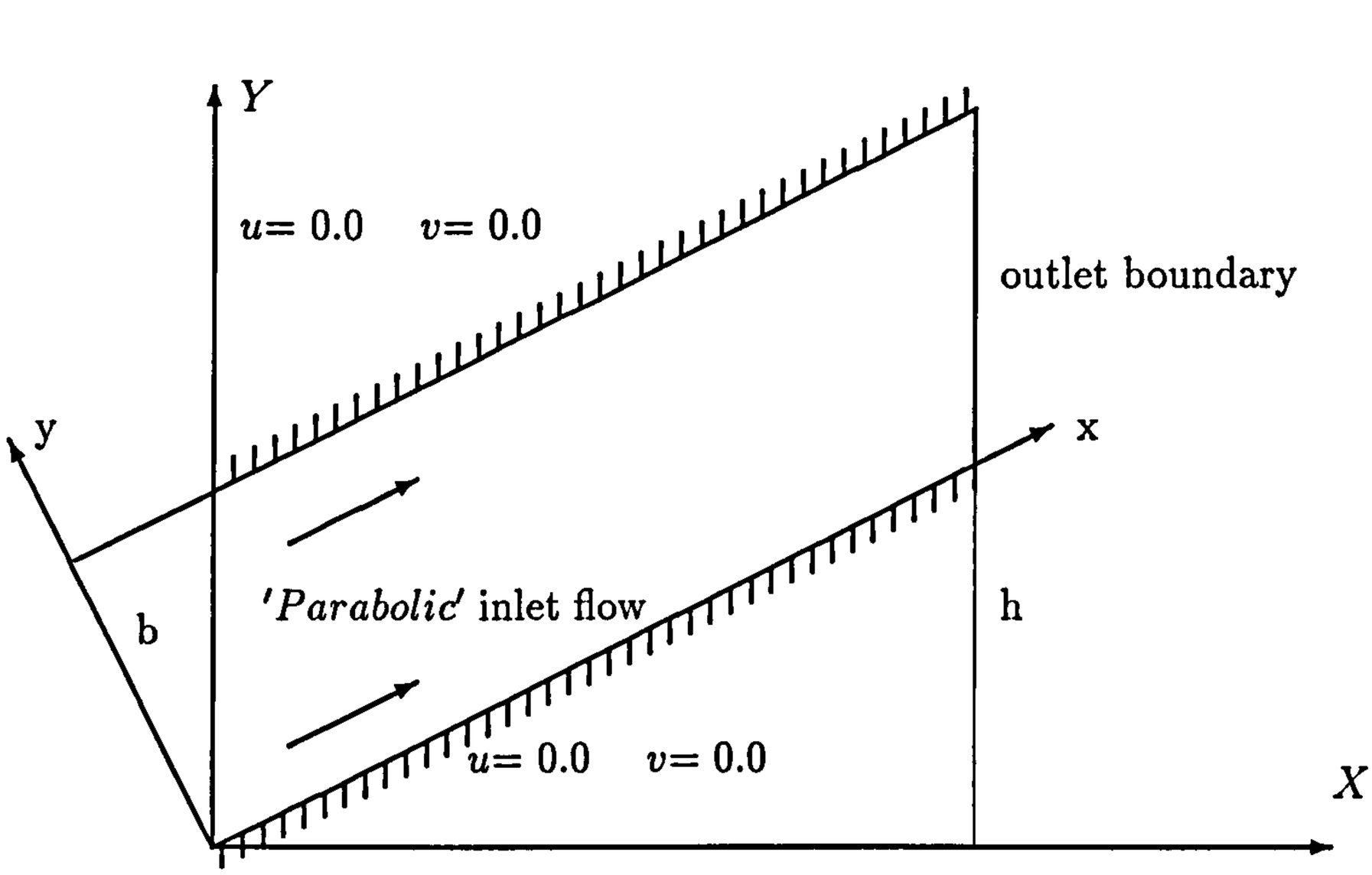


Figure 6.38: Picture to illustrate the channel flow - Non-coordinate axes aligned.

6.4.1 Basic Equations and Boundary Conditions

Because the flow is no longer aligned with the base coordinate axes, it is convenient to use the Cartesian coordinate system and place the flow along the x-axis, and then transfer the $x - y$ Cartesian coordinate system to the $X - Y$ Cartesian coordinate system. The fundamental equations for two dimensional incompressible flow are the continuity equation and Navier-Stokes equations. The equations used in solving the two-dimensional incompressible flow problems in rectangular coordinates are presented in Section 6.2.1 (Equations 6.1 – 6.3).

To investigate the channel flow, some special characteristics of the channel flow should be discussed before hand. Because the flow is fully-developed in the channel, we have:

$$v = 0$$

Substitute this on the Navier-Stokes equations become:

$$\frac{\partial u}{\partial x} = 0 \tag{6.51}$$

$$0 = g_x - \frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{\mu}{\rho}\frac{\partial^2 u}{\partial y^2} \tag{6.52}$$

$$0 = g_y - \frac{1}{\rho}\frac{\partial p}{\partial y} \tag{6.53}$$

The gravity can be resolved into the coordinate directions as:

$$g_x = -g\sin(\alpha)$$

$$g_y = -g\cos(\alpha)$$

Because $g_y = -g\cos\alpha$ and $\frac{dh}{dy} = \cos\alpha$, so

$$g_y = -g\frac{dh}{dy}$$

So Equations (6.51) and (6.53) yield:

$$u = u(y) \tag{6.54}$$

$$\frac{\partial}{\partial y}(\rho g h + p) = 0 \tag{6.55}$$

That is to say, the $\rho g h + p$ is a function of x. Now we investigate Equation (6.52),

$$g_x = -g \sin\alpha = -g\frac{dh}{dx},$$

where

$$\frac{dh}{dx} = \sin\alpha$$

The Equation (6.52) becomes:

$$\frac{d}{dx}(\rho g h + p) = \mu\frac{\partial^2 u}{\partial y^2} \tag{6.56}$$

6.4.2 Theoretical Solutions of the Channel Flow

Equation (6.56) is a second order, simple partial differential equation, we can integrate it directly. Integrating Equation(6.56), we have:

$$\mu\frac{\partial^2 u}{\partial y^2} = \frac{d}{dx}(\rho g h + p)$$

$$\frac{\partial u}{\partial y} = \frac{1}{\mu}\frac{d}{dx}(\rho g h + p)y + C_1 \tag{6.57}$$

Integrating this equation again, yields:

$$u(y) = \frac{1}{2\mu}\frac{d}{dx}(\rho g h + p)y^2 + C_1 y + C_2 \tag{6.58}$$

Constant factors C_1 and C_2 can be determined by the boundary conditions. In the xy coordinate system, the boundary conditions are:

$$u|_{y=0} = 0, \quad u|_{y=b} = 0,$$

So we have:

$$C_2 = 0$$

$$\frac{1}{2\mu}\frac{d}{dx}(\rho g h + p)b^2 + C_1 b = 0$$

$$C_1 = -\frac{1}{2\mu}\frac{d}{dx}(\rho gh + p)b$$

Substituting the values of c_1 and c_2 into equation (6.58), the velocity distribution along x axis is found to be given by,

$$u(y) = -\frac{1}{2\mu}\frac{d}{dx}(\rho gh + p)(by - y^2) \tag{6.59}$$

The distribution of the velocities have the parabolic shape in the non-coordinate axis aligned as we expected. In order to compare the theoretical solution with the numerical results, some numerical solutions of the theoretical analysis are needed. Except the velocity distribution along the x-direction, it is useful to investigate the change rate of the pressure along the middle of the channel from the above equation. Because the distribution of the velocity is parabolic, the maximum velocity will be created in the middle of the channel. The maximum velocity is:

$$\begin{aligned} U_{max} &= -\frac{1}{2\mu}\frac{d}{dx}(\rho gh + p)(b\frac{b}{2} - (\frac{b}{2})^2) \\ &= -\frac{1}{2\mu}\frac{d}{dx}(\rho gh + p)(\frac{b^2}{2} - \frac{b^2}{4}) \\ &= -\frac{1}{2\mu}\frac{d}{dx}(\rho gh + p)\frac{b^2}{4} \\ &= -\frac{b^2}{8\mu}\frac{d}{dx}(\rho gh + p) \end{aligned} \tag{6.60}$$

So the pressure drop along the middle of the channel is:

$$\frac{d}{dx}(\rho gh + p) = -\frac{8\mu}{b^2}U_{max} \tag{5.61}$$

we have:

$$\frac{dp}{dx} = -\frac{8\mu}{b^2}U_{max} - \rho g \sin\alpha \tag{6.62}$$

From the above equation, it is found that pressure drop is dependent only on the maximum velocity, the width of the channel, and the viscosity.

6.4.3 Theoretical Solution of Channel Flow With Non-aligned

In the previous section, we have derived the theoretical solution of the non-aligned laminar channel flow. The fundamental equations for two-dimensional incompressible flow are the Navier-Stokes equations. Given the special boundary conditions of the steady laminar channel flow, the analytical solution of the velocity distribution along x axis is:

$$U(y) = -\frac{1}{2\mu}\frac{dp}{dx}(by - y^2) \tag{6.63}$$

In order to compare the theoretical solution with the numerical results, the coordinate system of the analytical solution should be transformed to the Cartesian coordinate system that the numerical solutions have been used. The coordinate system is rotated anti-clockwise through an angle α to produce Oxy system and that P(X,Y) is a general point in this two coordinate systems (see Figure 6.39). We have the following equations to change the coordinates:

$$X = x\cos\alpha - y\sin\alpha$$

$$Y = x\sin\alpha + y\cos\alpha$$

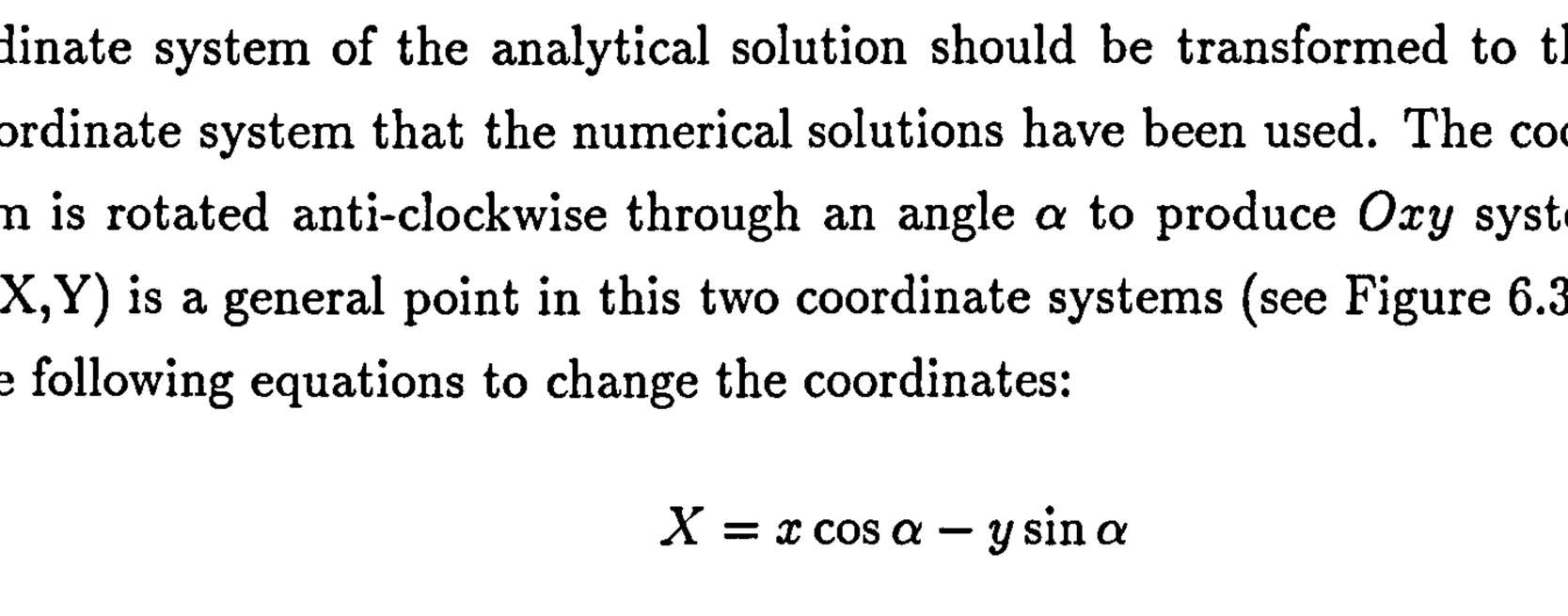


Figure 6.39: Relations of the two Cartesian coordinate.

Now we known that the values of the XY - coordinate, so the above equations should be changed to the rotated system, the Oxy - coordinate. The values of the xy - coordinate are derived from:

$$X \sin\alpha = x \cos\alpha \sin\alpha - y \sin^2\alpha$$

$$Y \cos\alpha = x \cos\alpha \sin\alpha + y \cos^2\alpha$$

The second equation minus the first, we have

$$X \sin\alpha - Y \cos\alpha = y(\sin^2\alpha + \cos^2\alpha)$$

so, the equation changes

$$y = Y \cos\alpha - X \sin\alpha$$

$$x = Y \sin\alpha + X \cos\alpha$$

The analytical solution is derived from $oxy-$ coordinate. It should be changed to the $OXY-$ coordinate, differentiate these equations, we have:

$$\frac{dX}{dx} = \cos,\alpha \qquad \frac{dY}{dx} = \sin\alpha$$

$$\frac{dp}{dx} = \frac{\partial p}{\partial X}\frac{dX}{dx} + \frac{\partial p}{\partial Y}\frac{dY}{dx}$$

substituting the above equations into the analytical solution of the laminar channel flow, the distributional velocity equation can be rewritten as

$$u(y) = -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)(by - y^2)$$

$$= -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)(b(Y\cos\alpha - X\sin\alpha) - (Y\cos\alpha - X\sin\alpha)^2)$$

The direction of the flow is along the x - axis in Oxy coordinate system, it should be transferred to the numerical calculation coordinate. The channel width b is changed to $b\cos\alpha$. The velocity equations of the channel flow in numerical calculation coordinate are:

$$u = -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)[b\cos\alpha(Y\cos\alpha - X\sin\alpha) -$$

$$(Y\cos\alpha - X\sin\alpha)^2]\cos\alpha \tag{6.64}$$

$$v = -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)[b\cos\alpha(Y\cos\alpha - X\sin\alpha) -$$

$$(Y\cos\alpha - X\sin\alpha)^2]\sin\alpha \tag{6.65}$$

This is the representation of the velocity equations of the channel flow in the computational coordinate system.

6.4.4 Numerical Results of Channel Flows

The laminar incompressible channel flow is a very simple example in fluid dynamics. It has well known results that can be used as the test bed for investigating the efficiency and accuracy of the numerical method. It is used to test the capabilities of the CC-PAMG code and extend its applications to other simulations, namely to develop a unfitted Cartesian based grid method with cut-cell method.

In the real calculation situation, the viscosity μ and the ratio of the pressure changes are to be assumed to be constant, and we choose this constant as 0.5 in our computing. The discretized Navier-Stokes equations have been used to calculate the results. The numerical computation is made with 288 cells to test the efficient and accurate of the Cartesian cut-cell method. The input data is displayed as in Equations (6.64) and (6.65). The Reynolds number is chosen as 100.

At this stage, we choose the zigzag channel boundaries to establish the computational domain, then cut the cells if the parts of these cells are out of the domain. The reference length in the x-axis direction is 6.0 and in y-axis direction it is 0.5. The actual channel length is $6.0/\cos(\alpha) = 6.184$, and the channel width is $0.50\cos(\alpha) = 0.485$.

The meshes for this case were generated by using the Cartesian cell grid with four levels of adaptive refinement. The two side conditions of the channel are fixed to zero for the OXY-coordinate system. At the outflow boundaries the velocity components

are extrapolated from the interior and the pressure is specified at its reference value. There are 288 coarsest cells illustrating the whole computation domain used to test this Cartesian cut cell method.

Although the Cartesian cut-cell has been introduced in this procedure, the visible output and display package has not been updated. It still displays the non-cut zigzag Cartesian mesh computational domain. Figure (6.41) shows the four-level constructions of the rectangular Cartesian meshes with the uniform refinement. Figure (6.42) shows the four-level constructions of the rectangular Cartesian meshes with the adaptive refinement, the third and fourth-level adaptive refinement have the partial refinement. The profiles of the four level velocities along the vertical lines are shown in Figure (6.43) which show a good agreement with the theoretical solution: they have the correct parabolic shapes. The profile of the theoretical solution is also displayed in this figure. Figure (6.40) shows the profiles of the streamline of the flow and the contours of the pressure distribution. Although the boundary meshes have not been cut out, the streamlines are located in the channel and the pressure distributions are zero outside the channel.

In the previous section, it is demonstrated that multigriding greatly improved the convergence rates - FAS multigrid procedures are very successful acceleration techniques when applied to the multigrid scheme. For the non-aligned channel flow, three-level computations using different relaxation sweeps after prolongation and restrictions are investigated. The convergence histories are shown in Figure (6.44) to Figure (6.47). In this case, the mesh refinement improves the solution through each level of refinement. However, the increased relaxation sweeps after prolongation and restriction do not appear to affect the convergence rates, especially in lower relaxation sweeps $\{F(4,2)$ and $F(2,2)\}$. Comparing the F(10,4) (relaxation sweeps 10 times after prolongation and sweeps 4 times after restrictions) cycles with F(2,2) cycles, the fourth level convergence rate of F(10,4) is nearly twice fast than the second level (35:63 work units), whilst the F(2,2) has the same convergence rate(60:64 work units).

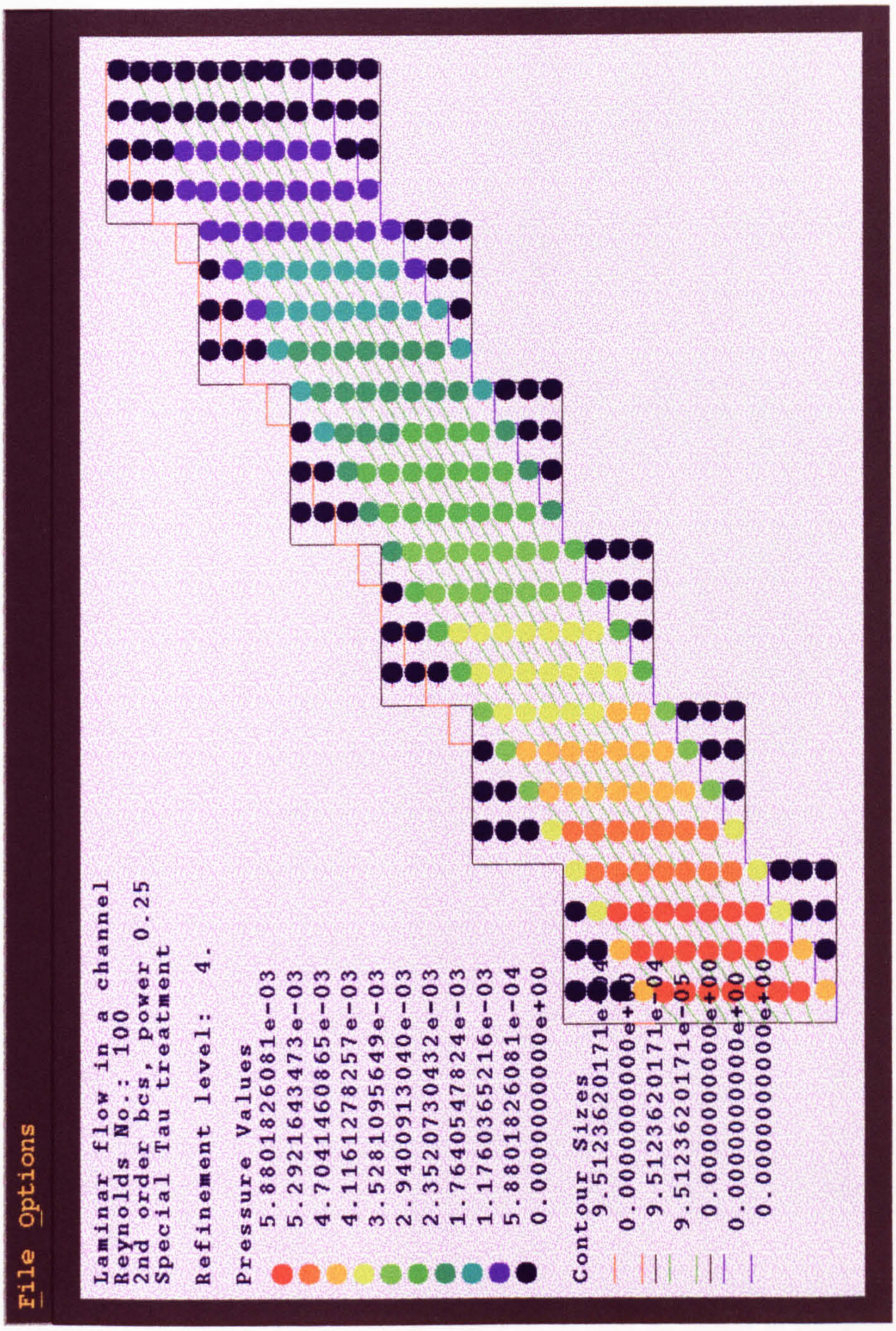


Figure 6.40: Non-aligned Channel Flow - Streamlines and Pressure Distributions for the Modified PAMG Solution at Reynolds Number = 100. The boundary meshes have not been cut out by using the Pview code.

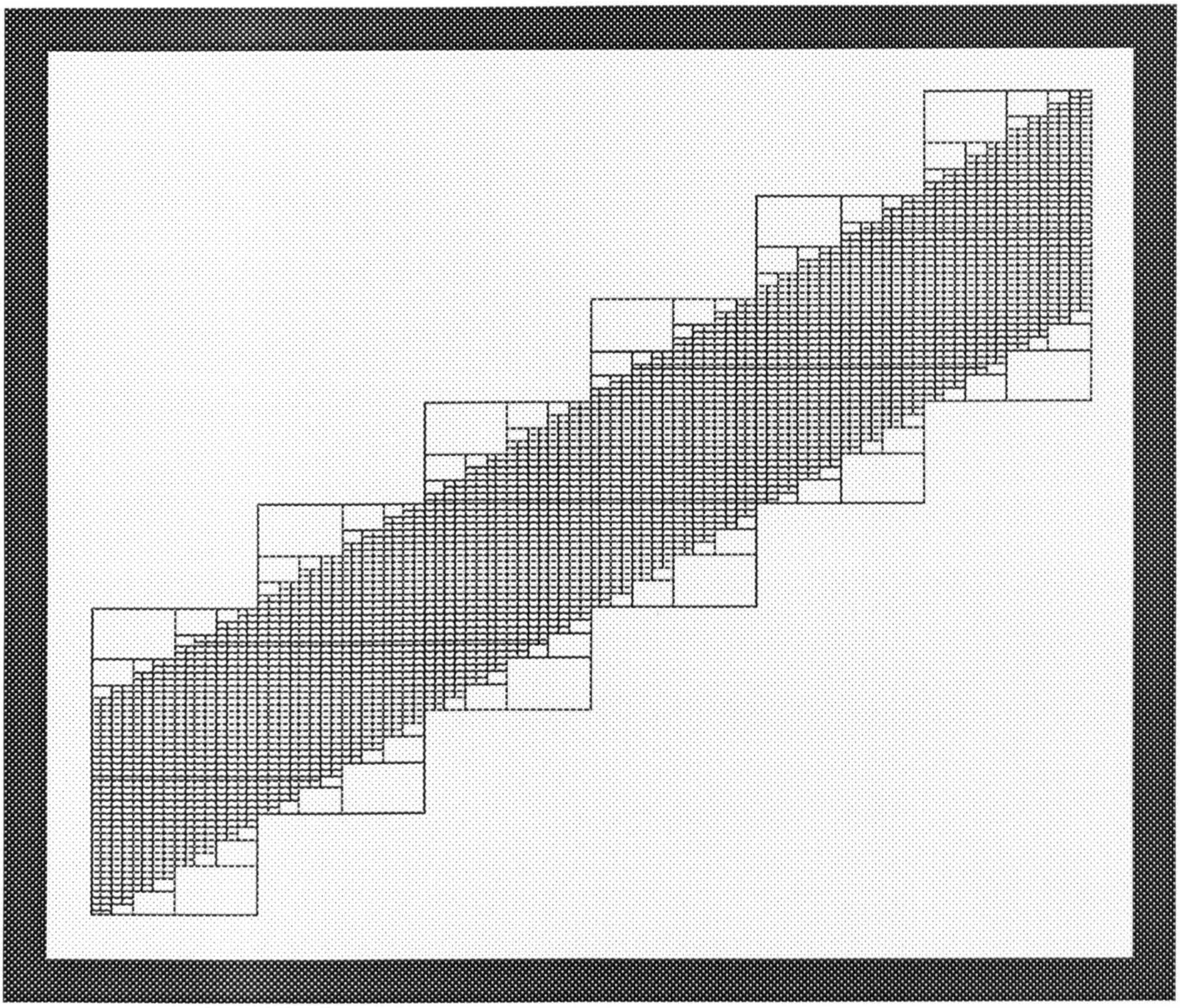


Figure 6.41: Non-aligned Channel Flows - Uniform Grid on Level 4 for the Modified PAMG Solution at Reynolds Number = 100. The boundary lines have not created by using Pview code, the meshes should be uniform but the computer made some mistake, it cannot give correct lines in this figure.

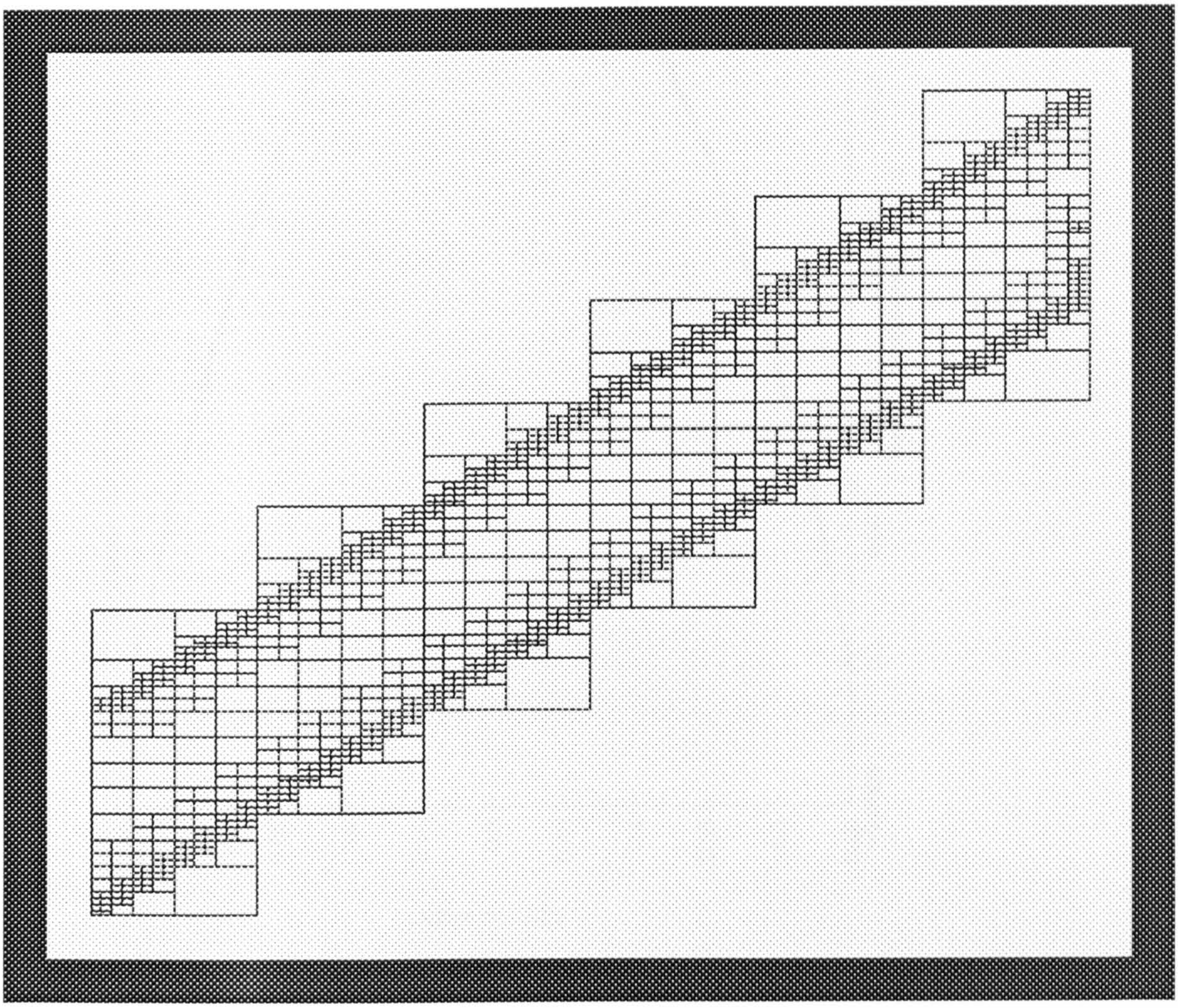


Figure 6.42: Non-aligned Channel Flows - Adaptive Grid Refinement on Level 4 for the Modified PAMG Solution at Reynolds Number = 100

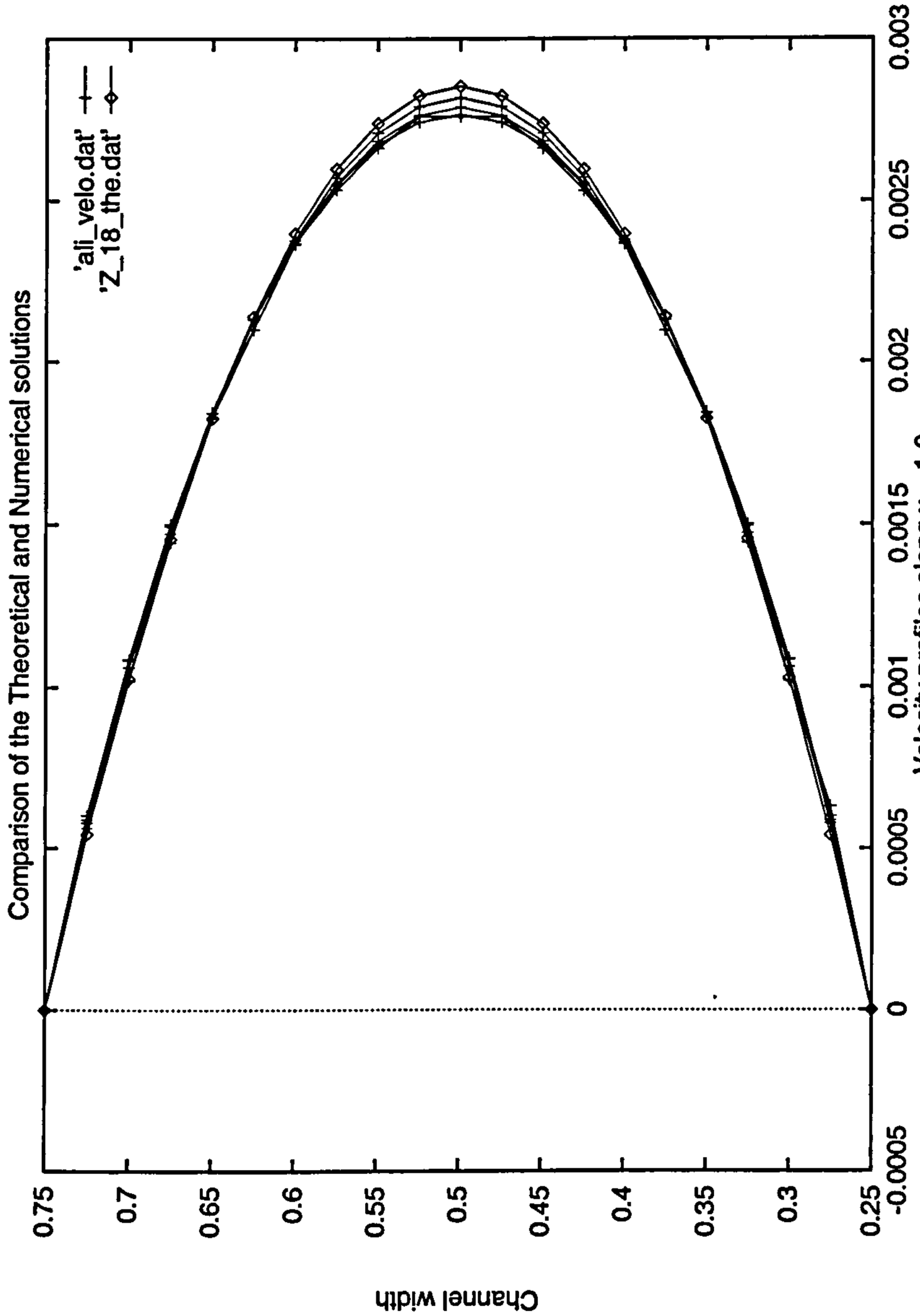


Figure 6.43: Velocity Profiles Along the Vertical Line (x = 1.0) of the Non-aligned Channel

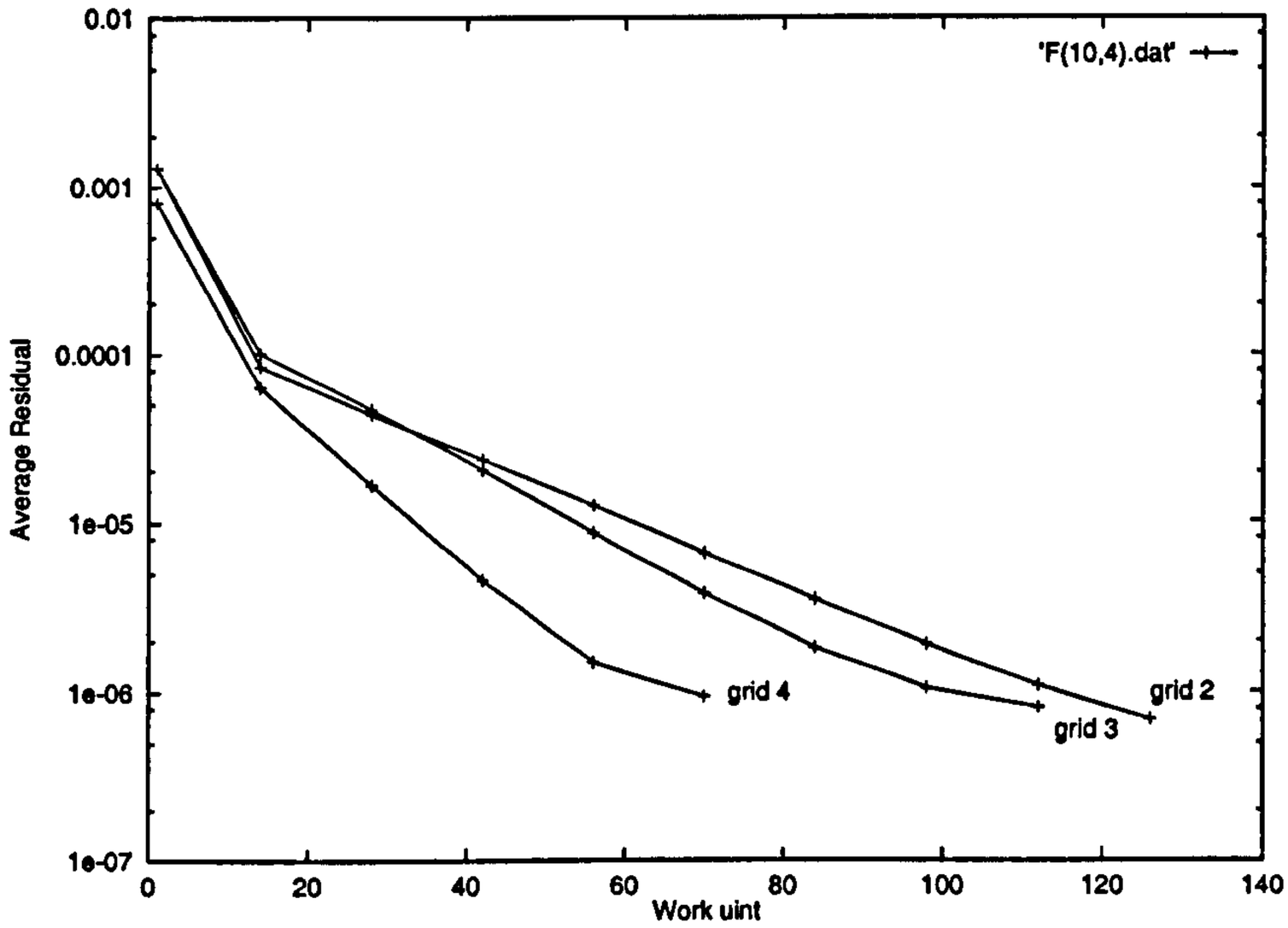


Figure 6.44: Non-aligned Channel Flow - Convergence of Solution for Uniform Multigrid Computations for Different Grid Levels at the Same Relaxation Sweeps After Prolongation and Restriction [F(10,4)]

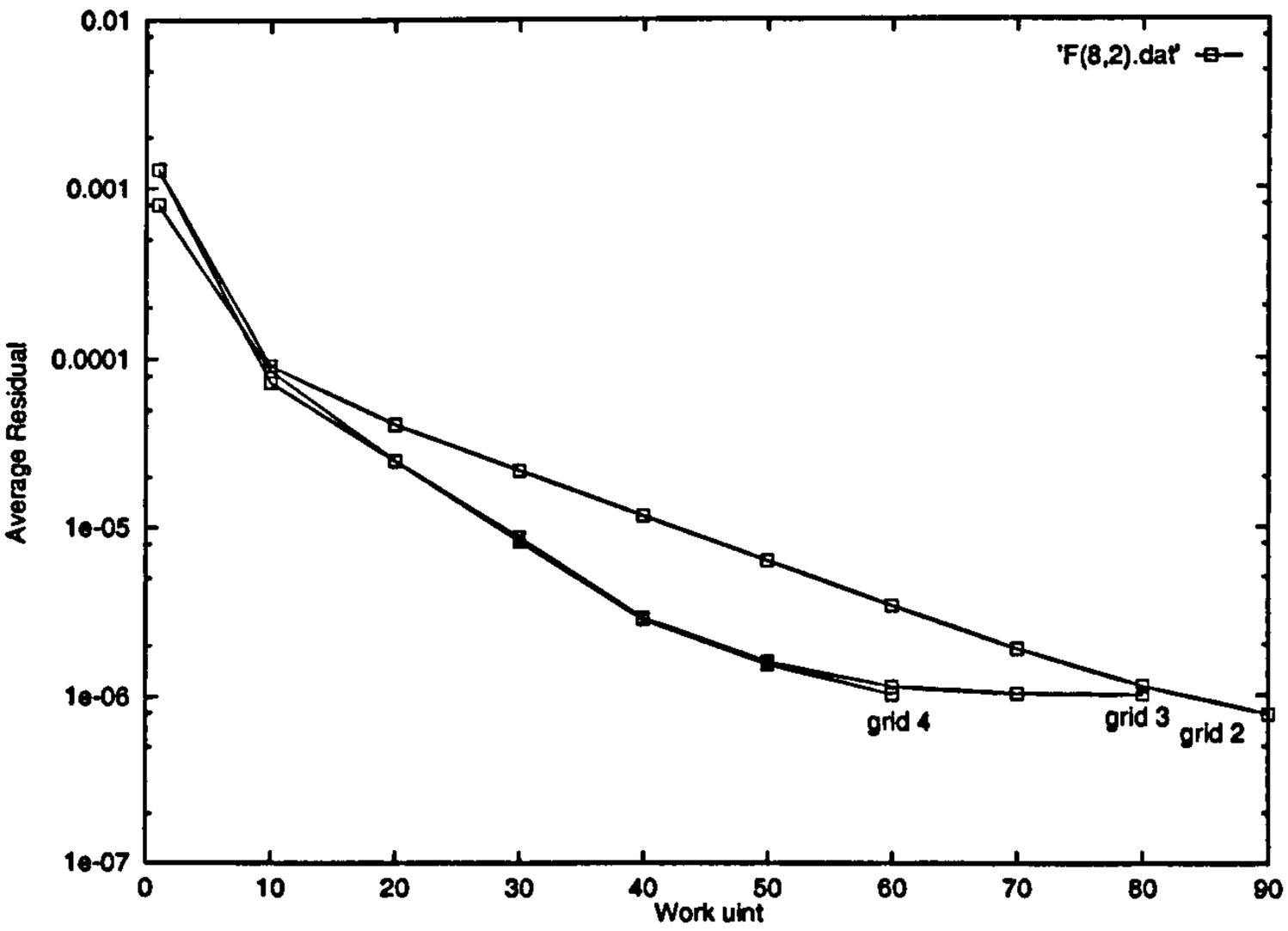


Figure 6.45: Non-aligned Channel Flow - Convergence of Solution for Uniform Multigrid Computations for Different Grid Levels at the Same Relaxation Sweeps After Prolongation and Restriction [F(8,2)]

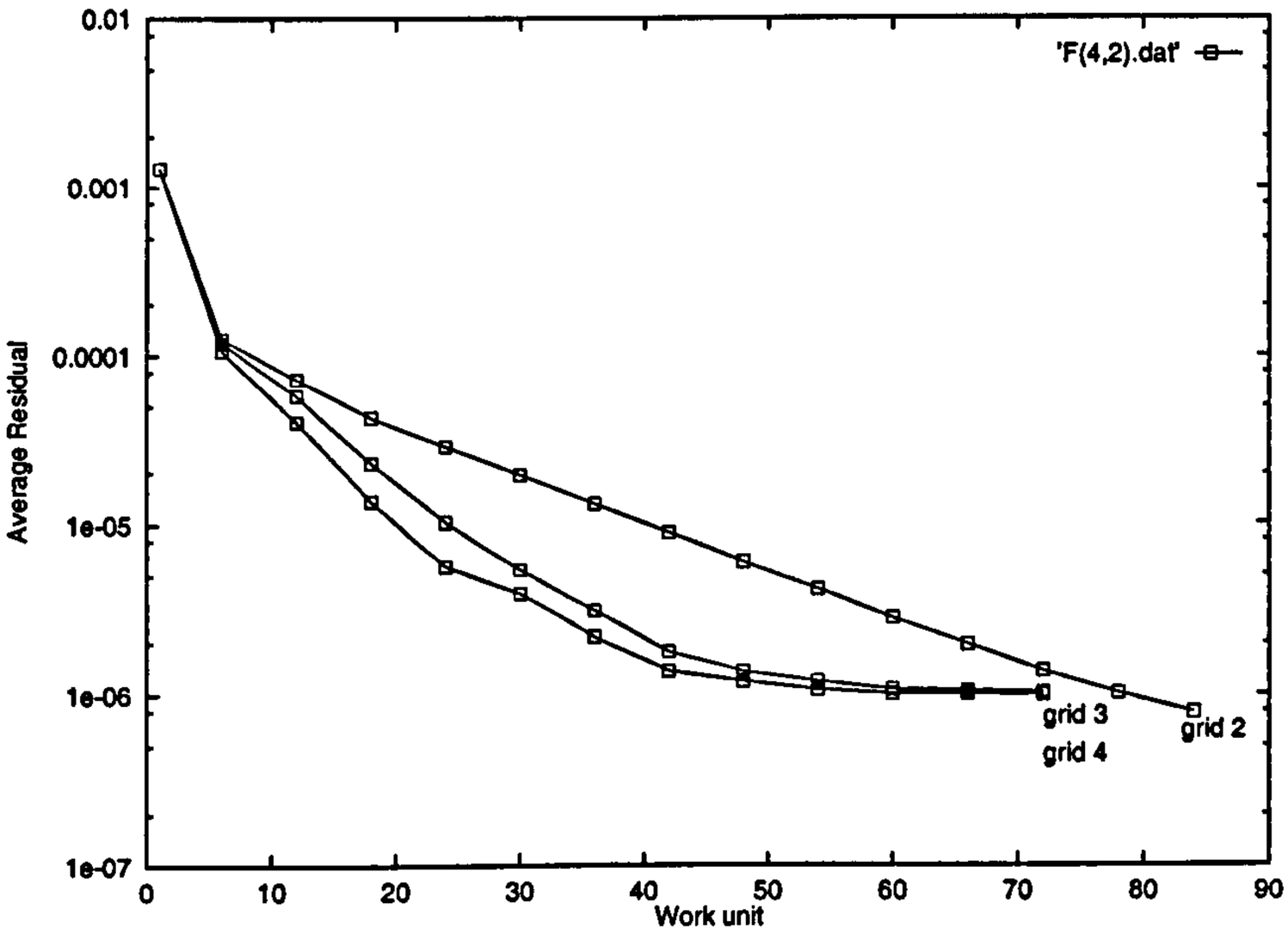


Figure 6.46: Non-aligned Channel Flow - Convergence of Solution for Uniform Multi-grid Computations for Different Grid Levels at the Same Relaxation Sweeps After Prolongation and Restriction [F(4,2)]

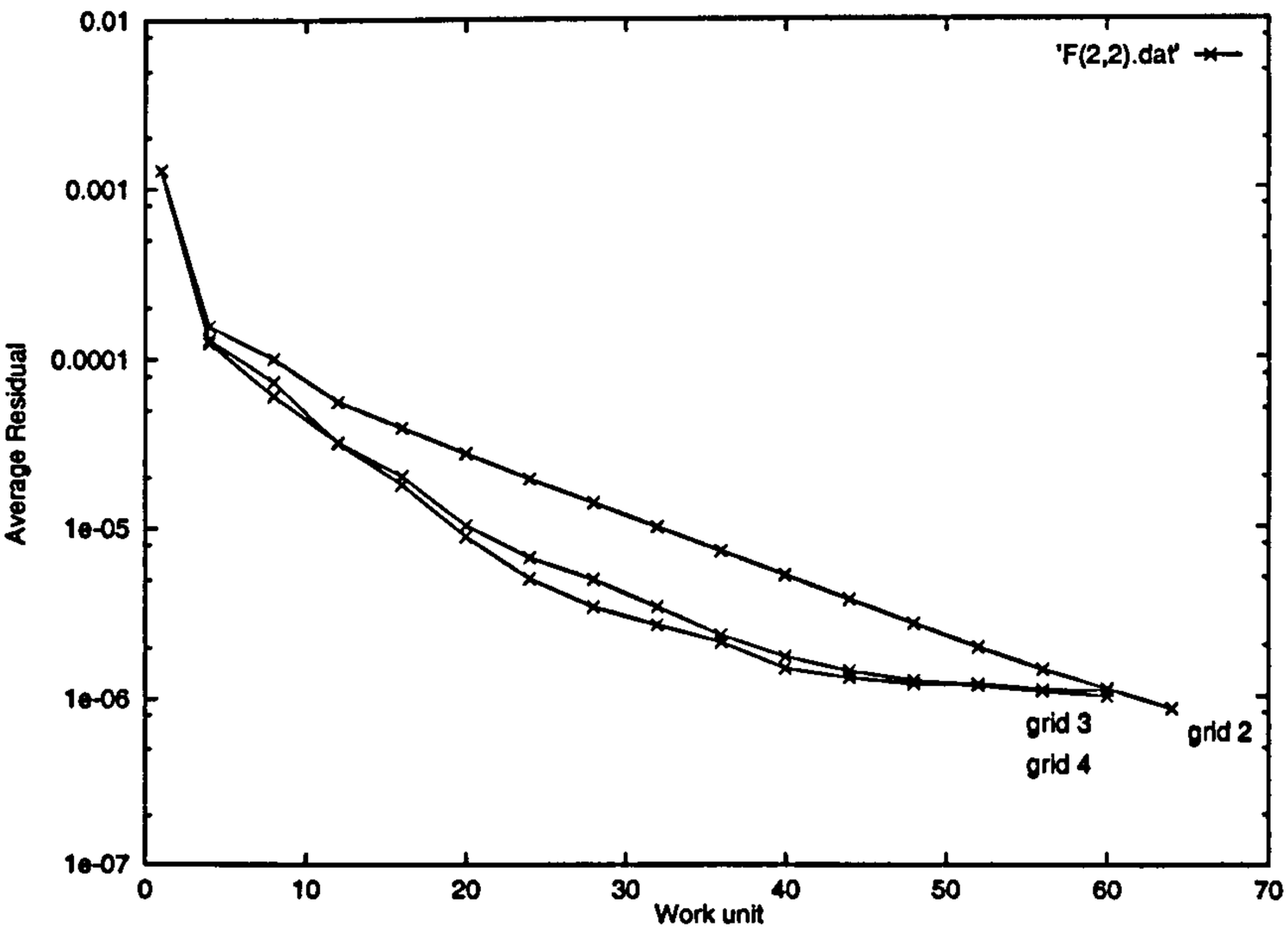


Figure 6.47: Non-aligned Channel Flow - Convergence of Solution for Uniform Multi-grid Computations for Different Grid Levels at the Same Relaxation Sweeps After Prolongation and Restriction [F(2,2)]

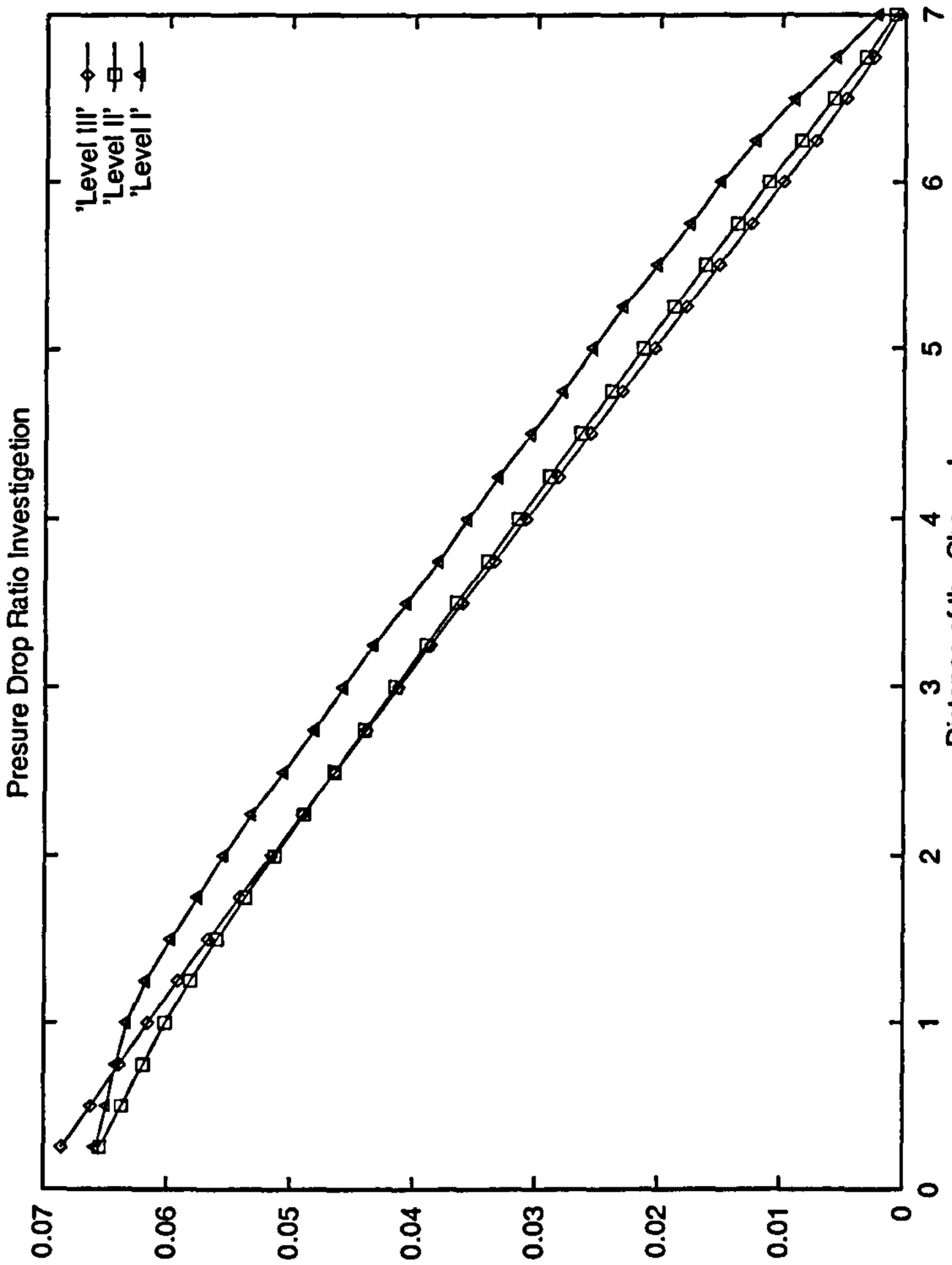


Figure 6.48: Three Different Pressure Histories Along the Middle of the Non-aligned Channel Flow

Effect of the rotation angle α for the error reduction

Next we consider the effect of the rotation angle α in the non-aligned laminar channel flow. There are several different rotation angles which have been investigated in the first quadrant part of the Cartesian coordinate system. It is simulated from the near x-axis direction to the near y-axis direction as five different steps. We start this simulation with the $\tan\alpha = 1/4$, then $\alpha = 1/2,\ 1,\ 2,\ 4$, respectively. The error reductions along the vertical line with the first 2 level grids are slightly changed as shown in table 6.7.

Rotated angle	Position	calculate Range	Error reduction (Err_1/Err_2)
14.0^0	x =1.0	$0.25 \sim 0.75$	3.892
26.6^0	x =1.0	$0.50 \sim 1.00$	3.297
45.0^0	x =1.0	$1.00 \sim 1.50$	2.688
63.4^0	y =1.0	$0.50 \sim 1.00$	3.10
76.0^0	y =1.0	$0.25 \sim 0.75$	3.921

Note: Calculated at x =1.0 or y=1.0 in OXY Coordinate System

Table 6.7: Error Reduction between grid 1 and 2 with the different angles

Table 6.7 gives the results of the error investigation of the non-aligned flow. The first column shows the angle α along the channel with the X-axis direction. The second column are the calculation position which computed along the vertical line or the horizontal line according the angle α. The third column shows the calculate ranges which changes with different angle α and the last column are the error reductions in different angles.

Effect of the Reynolds Number on error reduction

The effect of the angle α for the error reduction has been studied in previous section. The effect of the Reynolds number for the error reduction has also been investigated. The angle between the channel and the x-axis direction is selected to be 26.6^0,they

use the same physical calculation conditions, only the Reynolds number has been changed to test the error reduction with the mesh size. Table 6.8 gives the different error reduction results, the error reduction is slightly increased with the Reynolds number rising.

Reynolds Number	Error Reduction ($\frac{Err_{18}}{Err_{60}}$)
1.0	3.096
50.0	3.144
100.0	3.297
500.0	6.178

Note: Error Reduction between grid 1(Err_{18}) and grid 2 (Err_{60})

Table 6.8: Error reduction with the different Reynolds numbers

6.4.5 Comparison of Theoretical Solution with Numerical Results

- `Theoretical solution of pressure drop`

In order to compare the theoretical solution with the numerical results, there are initial problems to need solved. In the theoretical solution, the maximum velocity in the middle of the channel is a function of the pressure drop, channel width, and the viscosity. In order to get the pressure drop, the numerical solution of the maximum velocity is needed. From CC-PAMG initial data, the maximum velocity is:

$$\begin{aligned} U_{max} &= 0.5(by - y^2) \\ &= 0.5(\frac{b^2}{2} - \frac{b^2}{4}) \\ &= \frac{b^2}{8} \end{aligned} \tag{5.66}$$

The pressure change ratio along the channel is

$$\begin{aligned}\frac{dp}{dx} &= -\frac{8\mu}{b^2}U_{max} \\ &= -\frac{8\mu}{b^2}\frac{b^2}{8} \\ &= -\mu \end{aligned} \tag{5.67}$$

where μ is:

$$\mu = \frac{\rho b}{Re}U_{max} \tag{5.68}$$

so the pressure drop along the middle of the channel is,

$$\begin{aligned}\frac{dp}{dx} &= -\frac{\rho b}{Re}U_{max} \\ &= -\frac{\rho b}{Re}\frac{b^2}{8} \\ &= -\frac{\rho}{Re}\frac{b^3}{8} \end{aligned} \tag{5.69}$$

To compare the pressure drop of the theoretical solution with the PAMG results, the channel width and the initial velocity must be set the same. In the numerical computation, the channel width is $2\cos(\alpha) = 1.7889$, the Reynolds number is 100, ρ is chosen as 1, so the pressure drop in theoretical solution is:

$$\frac{dp}{dx} = -\frac{\rho b^3}{800} = -\frac{5.7248}{800} = -7.155965E10^{-3}$$

- `Numerical results of the pressure drop`

Due to the irregular boundaries and non-aligned channel flow, the calculated pressure in the middle of the channel is no longer along the X direction, but only a number of points calculated along the middle of the channel. Figure 6.48 shows the three level pressures along the middle of the channel and the numerical results of the pressure drops are shown in Table 6.9. The pressure drops along the middle of the channel are obtained by dividing the channel with 27 steps, and calculating the local pressure drops, then adding them together and averaging to obtain the whole pressure drop along the middle of the channel. The formula that we used is given as following:

$$\left.\frac{dp}{dx}\right|_{middle\_of\_channel} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{P_{i+1} - P_i}{x_{x+1} - x_i}\right)$$

Grid Level	No. of cells	Pressure drop ratio
1	288	-8.426E-3
2	960	-8.561E-3
3	3456	-8.949E-3

Table 6.9: Pressure Drop Proportions Along the Middle of the Channel with proper mesh refinement

Comparing the results for the pressure drop ratios, it is evident that the pressure drops have decreased linearly in a constant factor, although they have the distinguished difference. The pressure drop ratio in level 1 is the nearest approach to the theoretical result and the pressure drop rate in level 3 is the worst result comparing to the other two numerical solutions.

Numerical Solutions of Velocity

In Chapter 4, we estimated the magnitude of the discretisation error as a function of the mesh lengths by using the Richardson Extrapolation method. To compare the theoretical solution with the numerical results, we consider the theoretical and numerical flow velocities in a certain computing domain. Comparing the theoretical solution with these numerical results, three different errors have been calculated in certain calculation positions.

Pember et al. [95] performed a convergence study to use the following equations to calculate the global error and irregular boundary errors:

$$error = \sum_{\Lambda_{ij}>0} \left| U^e_{ij} - U^c_{ij} \right| \Lambda_{ij} \Delta x \Delta y$$

$$Wall_{error} = \sum_{\Lambda_{ij}>0, \Delta_{ij} \in bndry} \left| U^e_{ij} - U^c_{ij} \right| \Lambda_{ij} \Delta x$$

where U^e and U^c are the exact and the computed solutions in the same positions,

respectively. Let Λ_{ij} is the volume fraction of the computational cell $\Delta_x \times \Delta_y$ that is inside the flow domain. If Λ_{ij} is equal to 1, this cell is inside the calculated domain; if Λ_{ij} is equal to zero, this cell is located outside the calculated domain; If $0 < \Lambda_{ij} > 1$, this cell is the irregular cell which is located on the boundary.

Finest Grid	Δx	Δy	Global Error	reduction ratio
1	0.5	0.125	4.143E-4	
2	0.25	0.0625	2.046E-4	4.0495
3	0.125	0.03125	1.034E-4	3.736

Table 6.10: Error investigation in the considered domain (between $2.0 < x < 4.0$)

The error calculations are tabulated in Table 6.10 for three uniform refinement levels. The theoretical solutions in the non-aligned channel are calculated at the same locations as the numerical results. In order to take out the effects of the inlet and outlet regions, the inlet part and the outlet part of the channel flow are not considered to calculate the error reductions. The calculation domain is located between $2.0 < x < 4.0$ in the non-aligned channel. The calculation errors along the streamline are computed and then added together. The different numerical errors and reduction ratios along the streamlines are shown in Table 6.10.

The error reduction investigations are computed in this way: in order to compare the first level error with the second level error, the same position must be met. For example, there are 4 calculation positions in first level and 8 in the second level. So the position in the second level should be calculated into 4 positions as in the first level. The first two positions in second level averaged to meet the first calculation position of the first level. Using the same method, the error reduction results are shown in last column of Table (6.10). This demonstrates that the CC-PAMG algorithm not only has an efficient error reduction as the mesh is refined, but also demonstrates that the numerical approximation is approximately second-order accurate.

6.4.6 Discussion

From this investigation, it seems reasonable to assume that this error will usually decrease as the mesh lengths are reduced. Comparing the reduction of errors in velocity, we found that the errors are decreased with mesh lengths, especially when α is small, where the error reduction approach to the theoretical solution. It is shown that the sequence of solutions obtained using finer and finer meshes will eventually give a solution that differs from its immediate predecessor by less than some assigned tolerance. We have demonstrated that we can achieve mesh independence for this simple, non-aligned geometry.

After careful consideration of this problem, we found that the error reduction near the x-axis and y-axis directions is larger than the others. At the angle α equal 45^0, the error reduction of the refinement is the smallest one. There are some reasons to affect this result. One is that the channel width is slightly changed when the angle α increase from the x-axis to y-axis direction.

It can be shown that the theoretical solution of the non-coordinate channel flow is:

$$u = -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)[b\cos\alpha(y\cos\alpha - x\sin\alpha) - (y\cos\alpha - x\sin\alpha)^2]\cos\alpha$$

$$v = -\frac{1}{2\mu}\left(\frac{\partial p}{\partial X}\cos\alpha + \frac{\partial p}{\partial Y}\sin\alpha\right)[b\cos\alpha(y\cos\alpha - x\sin\alpha) - (y\cos\alpha - x\sin\alpha)^2]\sin\alpha$$

From these equations, we assume that the viscosity μ and the channel width are constant. The viscosity μ is constant which means the average Reynolds number is $Re = \frac{\rho L u_{ave}}{\mu}$. In the numerical computation, the channel width is changed with the rotation angle α, the viscosity μ should be changed with different channel widths to keep the Reynolds number constant. Actually the Reynolds number is constant in

this procedure. There are some errors that are introduced to the numerical results. It causes the error reduction to become smaller.

Another possible reason is the method of calculating the solution, which requires interpolation formulas. From the present study, the velocity is calculated along the vertical line (x=1.0) or horizontal line (y=1.0). When the channel lies near either the x or y-axis directions, the velocity distributions along the vertical line or the horizontal line are similar to the velocity along the vertical line of the channel. At $\alpha = 45^0$, the two velocities have significant differences. This may affect the results in Table 6.7. It is also clear that the procedure is far more accurate in high Reynolds number flows.

6.4.7 Conclusion

A method for the computation of steady incompressible channel flow involving a non-aligned channel has been presented. Based on a Cartesian cut-cell approach and a high resolution finite volume scheme with adaptive multigrid method, the present method proceeds in two steps for the calculation. Firstly, a fixed background Cartesian mesh is constructured on the computational domain. Using this, the Cartesian cut-cell approach can be seen as a viable alternative to fitted mesh methods for coping with extremely complicated geometries. Secondly, the adaptive refinement method can be used to refine the actual computational domain to have the more accurate and effective numerical results.

The computed results suggested that the present approach can provide an accurate and effective numerical prediction tool for the simulation of steady incompressible flow fields involving complex geometry boundary conditions. Further validation of this method for more complex geometry is discussed in the next Chapter.

Chapter 7

Flow round a Curved Channel

7.1 Introduction

Chapter 6 highlighted the potential problems of applying the Cartesian cut-cell method to several typical problems in fluid dynamics. We have investigated the solutions for the laminar flow along a semi-infinite flat plate, derived the theoretical solutions and computed the numerical results. Although the two algorithms (PAMG and CC-PAMG) have essentially the same results in the backward-facing step flow, the CC-PAMG has faster convergence ratio than the original PAMG. In non-aligned channel flow, theoretical solutions are compared with the numerical results. The pressure drop ratio along the middle of the channel, error reduction with the grid levels, and convergence rate are also investigated. It is demonstrated that our new CC-PAMG code is a more efficient and accurate algorithm.

In order to apply this algorithm to more complex geometry and demonstrate that the proposed Cartesian cut-cell method is suitable for any complex geometry conditions in two-dimensional flow, information on more complex geometry is required. We have considered curved channel flow and flow around a semi-circular bend to investigate the performance of the complex boundary conditions in two-dimensional flow.

7.2 Laminar Flow Around a Curved Channel

There is no theoretical solution for the curved channel flow which can be very complicated in the bent part of the channel. The average distribution across the flow is no longer uniform and can be difficult to predict. The correctness of the results is established by comparing CC-PAMG with other codes rather than experiments. The reference code is a commercial CFD code, CFX 4.2 [57], which has also been extensively validated and is widely used by many users. CFX 4.2 solves the same calculation domain, using a different discretisation. Although using the multi-block structured method to create the computational domain, the finite difference approach, using the discrete points, associated with boundary-fitted grids, is still used in CFX code. The disadvantages of the structured grid are still exist in it.

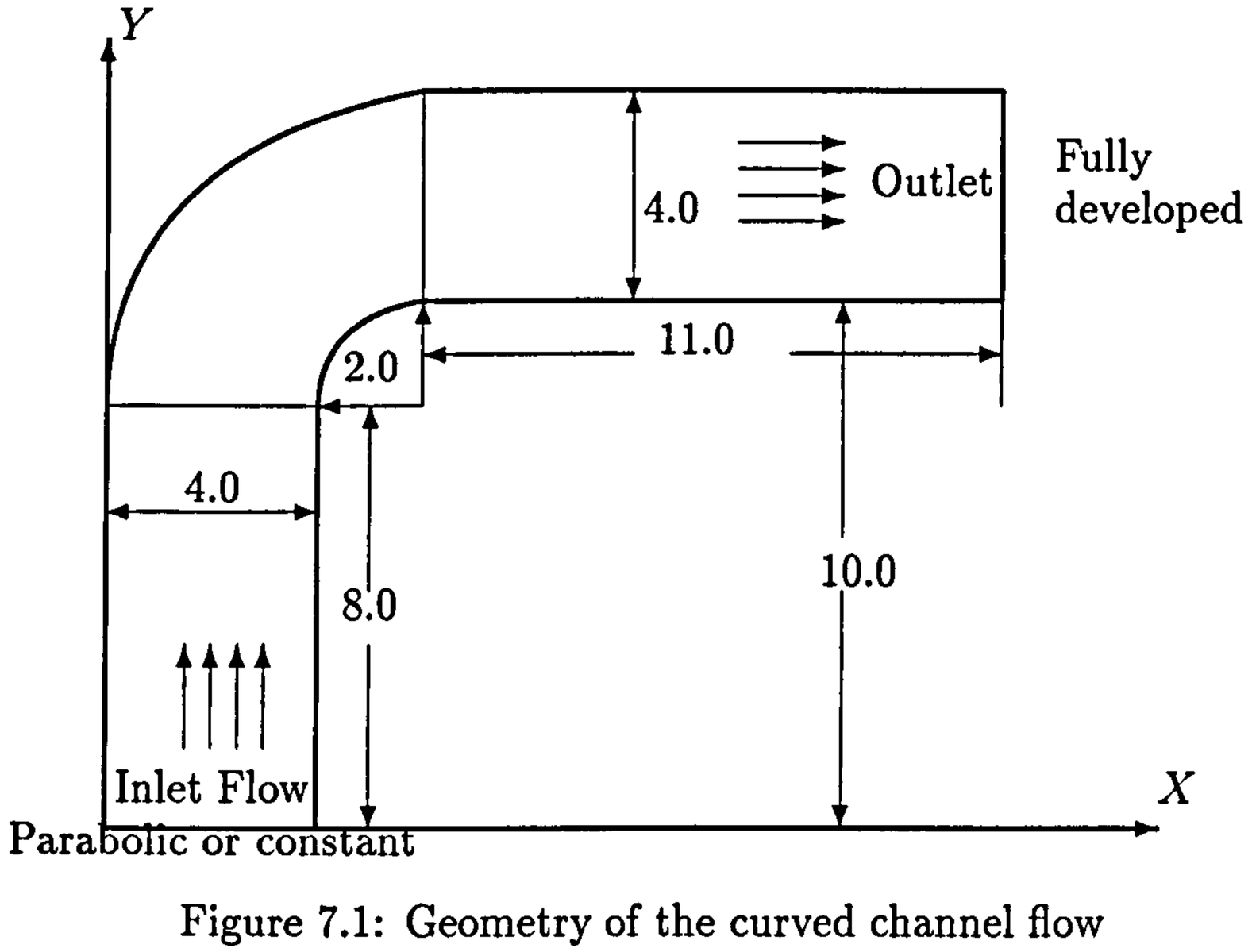


Figure 7.1: Geometry of the curved channel flow

For both inlet profiles, the geometry of the calculated domain is indicated in Figure 7.1. The channel length is 8.0 units before the bend, the width remains constant

before, during, and after the curvature which is 4.0 wide. The channel is bent in a right-angle near the middle of the channel. After the curvature, there is an exit section 11.0 units long and the total equivalent length of the channel is 25.3 units.

Different values for the Reynolds number have been used in these computations, give values of Re, typically $50 \leq Re \leq 500$. The input velocity profiles for these flow rates were uniform and parabolic. This profile is compatible with a long entry duct and a fully-developed flow profile. The effects of using a flat profile were to decrease the potential for separation due to the increase energy around the curve and to compare with the commercial package - CFX 4.2. The fully developed parabolic inlet profile was used to investigate the convergence performance with CC-PAMG algorithm.

The two different inlet flows are investigated in the same calculation domain. No-slip boundary conditions are applied on all the walls. In this configuration, we will calculate the flow upstream of the bent channel, thereby solving the flow along the channel especially in the curvature and after the bend. The important features of this flow are the bend zone and the velocity profiles downstream of the bent channel which we discuss later in this section. The solutions for the pressure drop along the middle of the bend channel are investigated and compared with CFX 4.2 results.

7.2.1 Numerical solution for Bent Channel Flow

A coarse base grid is generated with the Cartesian cut-cell approach and the scheme converges to a solution through the requested 4 levels of adaptive mesh refinement with tolerance 10^{-6}. Figure 7.2 shows the streamlines computed from the CC-PAMG solution. The velocity vector plot is given in Figure 7.5 (CFX 4.2 results). This depicts the calculated development of laminar flow, showing an increase of velocity in the inner regions at the entrance and during the initial stages of the bend. The bent zone, which characterises this type of flow, appears clearly since the velocity vectors change gradually from the vertical direction to the horizontal direction (Figure 7.5,

CFX 4.2 solutions), and the streamlines smoothly pass through the curved zone (Figure 7.2, CC-PAMG solutions). A close-up of the adapted grid for the bended channel at the third level and the fourth level of mesh refinements is shown in Figure 7.3 and Figure 7.4 respectively. Figure 7.10 shows the improvement of the solutions due to mesh refinement, at a given location just after the bend. Both schemes (CC-PAMG and CFX 4.2) are compared to each other at a selected series of locations in Figure 7.8 to Figure 7.11. Note that the aspect ratio of the vertical and horizontal distances is not in scale on these figures.

The boundary geometries in curved zone are displayed as steps rather than a smooth curved boundary. The reason is the Post-processor PVIEW which we used is designed to display only rectangular boundary conditions, and is unable to display more complex geometry conditions. So Figures 7.2, 7.3 and 7.4 have zigzag boundaries in the curved zones. Although the Cartesian cut-cell that was introduced in Chapter 4 is cut by straight lines not the curved lines, that the curved boundaries are approached by a series of straight lines, the calculated solutions along the curved boundaries have little disturbance by the straight lines approaching.

Figure 7.6 illustrates how the flow velocity changes gradually from the vertical to the horizontal direction when passing through the right-angle bend. Notice that some parts of the bent channel wall are slightly disturbed due to the modified polygon clipping algorithm. However, the velocity near the wall is relatively small, the un-smoothed bent channel wall have little effect on the velocity profiles. The velocity profiles near the wall are nearly smooth. Figure 7.7 shows that the velocity profiles change gradually from the vertical direction (the velocity u is relatively small before entering the bend) to the horizontal component of the velocity which the vertical component v is small after the bend. The blue colour indicates the vertical velocity v, whilst the green indicates the horizontal component u in Figure 7.7. It is clearly shown that the horizontal component u exceeds the vertical component v after the 45^0 degree bend.

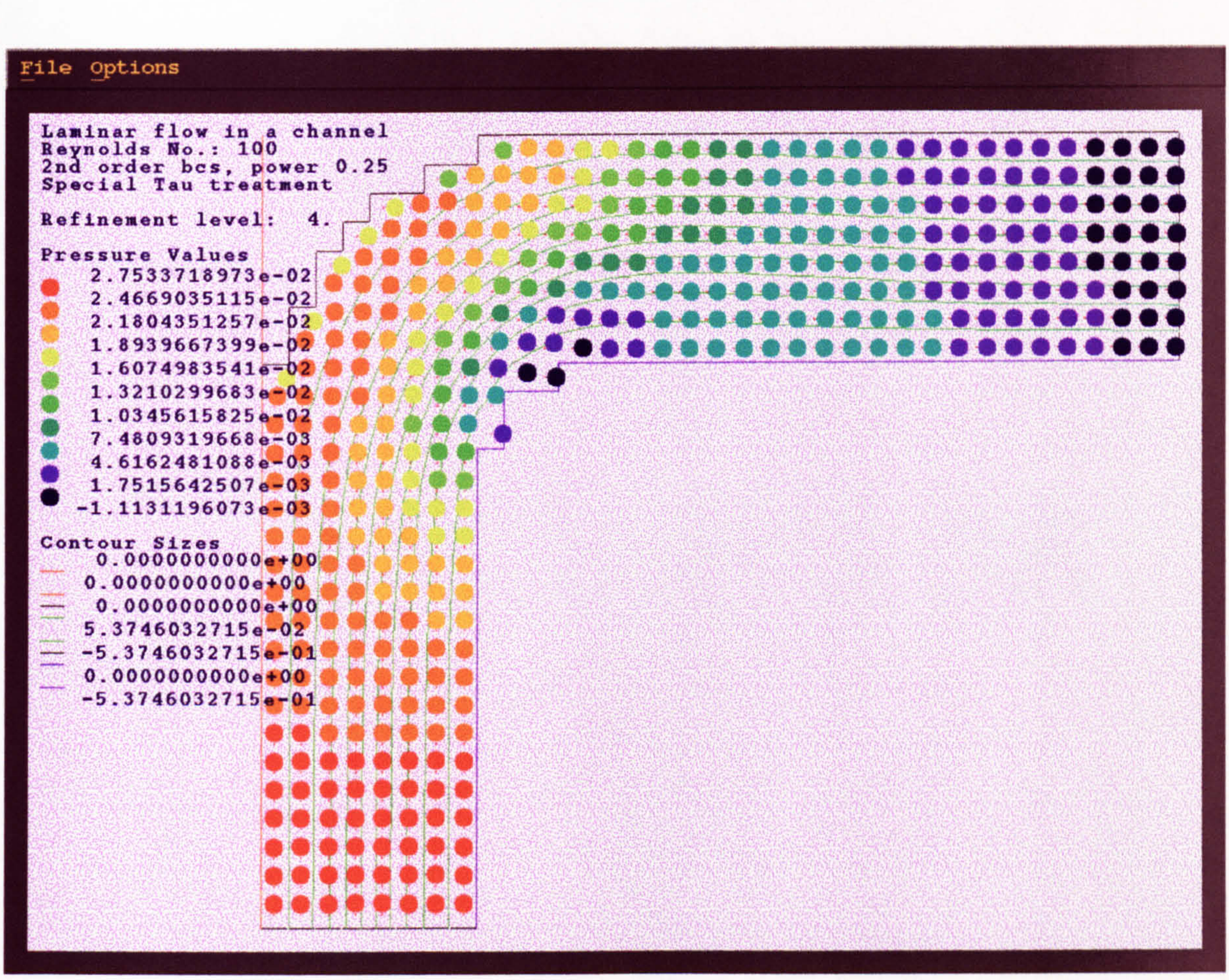


Figure 7.2: Curved channel flow problem - Streamlines and pressure distributions for the CC-PAMG solution at Reynolds number = 100

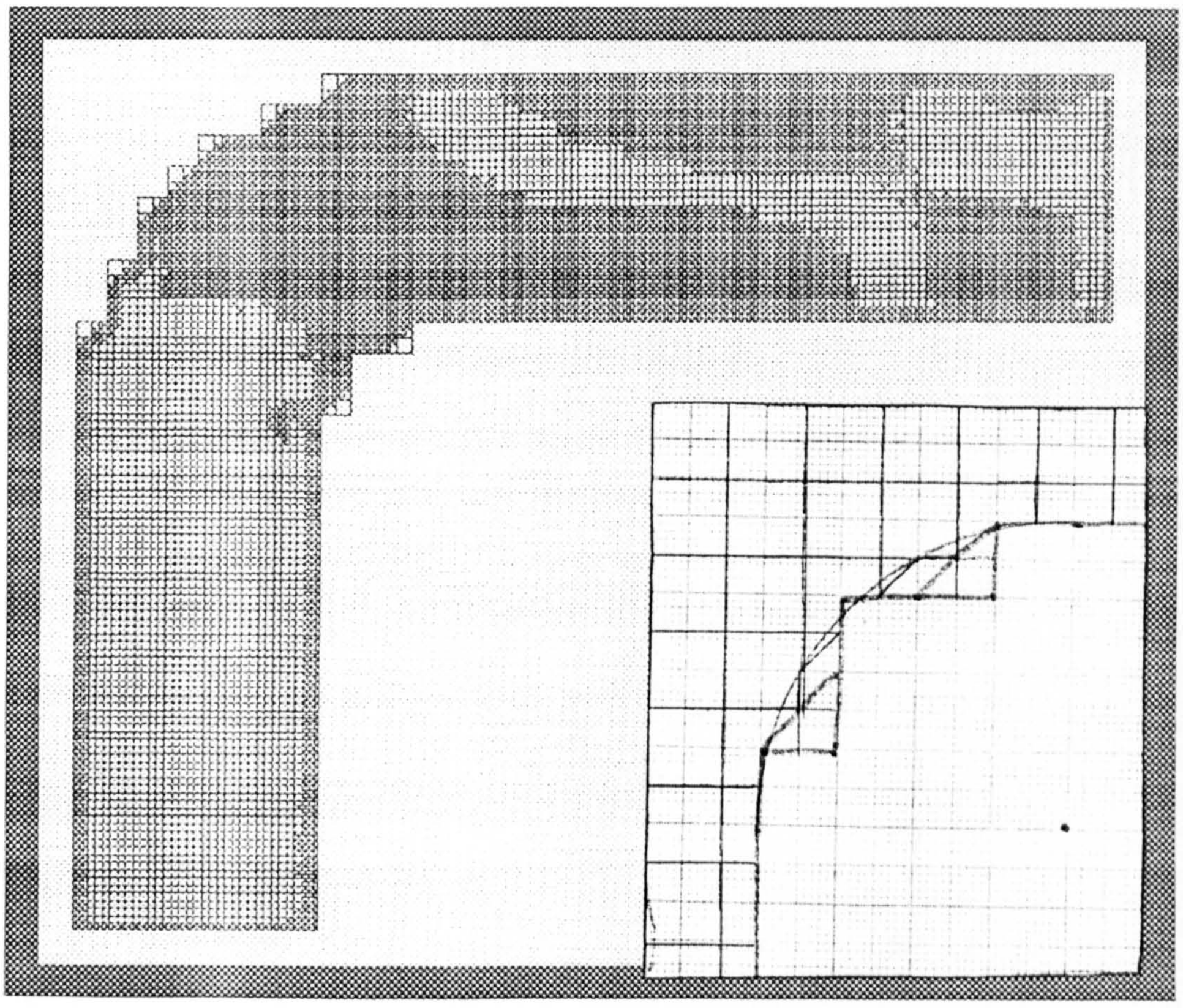


Figure 7.3: Curved channel flow problem - Diagram of adapted grid on an level 3 for the CC-PAMG solution at Reynolds number = 100. Actual cell structures used for calculation of changes on the inside bend boundary are shown in the right blow.

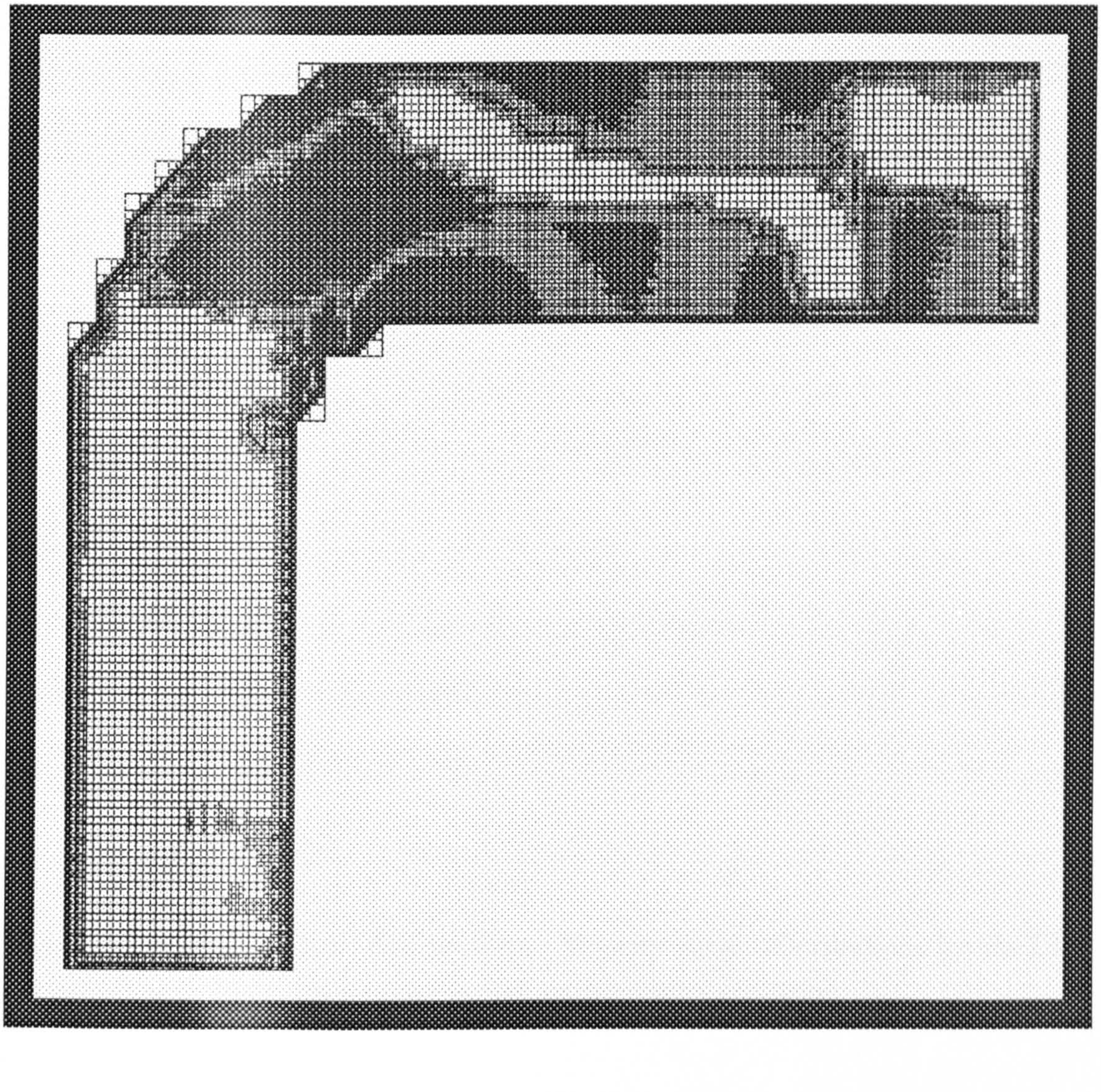


Figure 7.4: Curved channel flow problem - Diagram of final adapted grid on an level 4 for the CC-PAMG solution at Reynolds number = 100

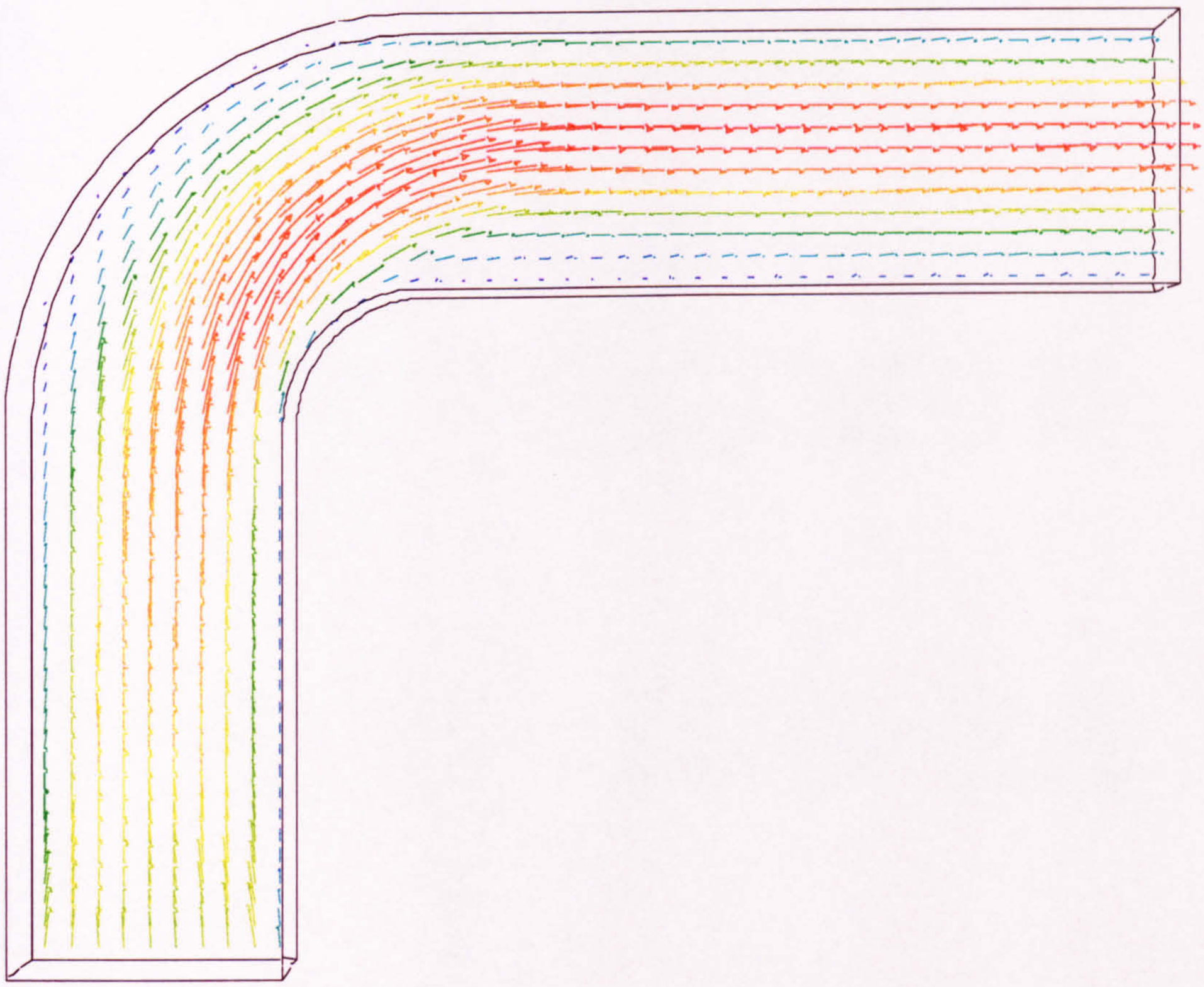


Figure 7.5: Curved channel flow problem - Velocity vector distributions for the CFX 4.2 solutions at Reynolds number = 100, uniform inlet condition

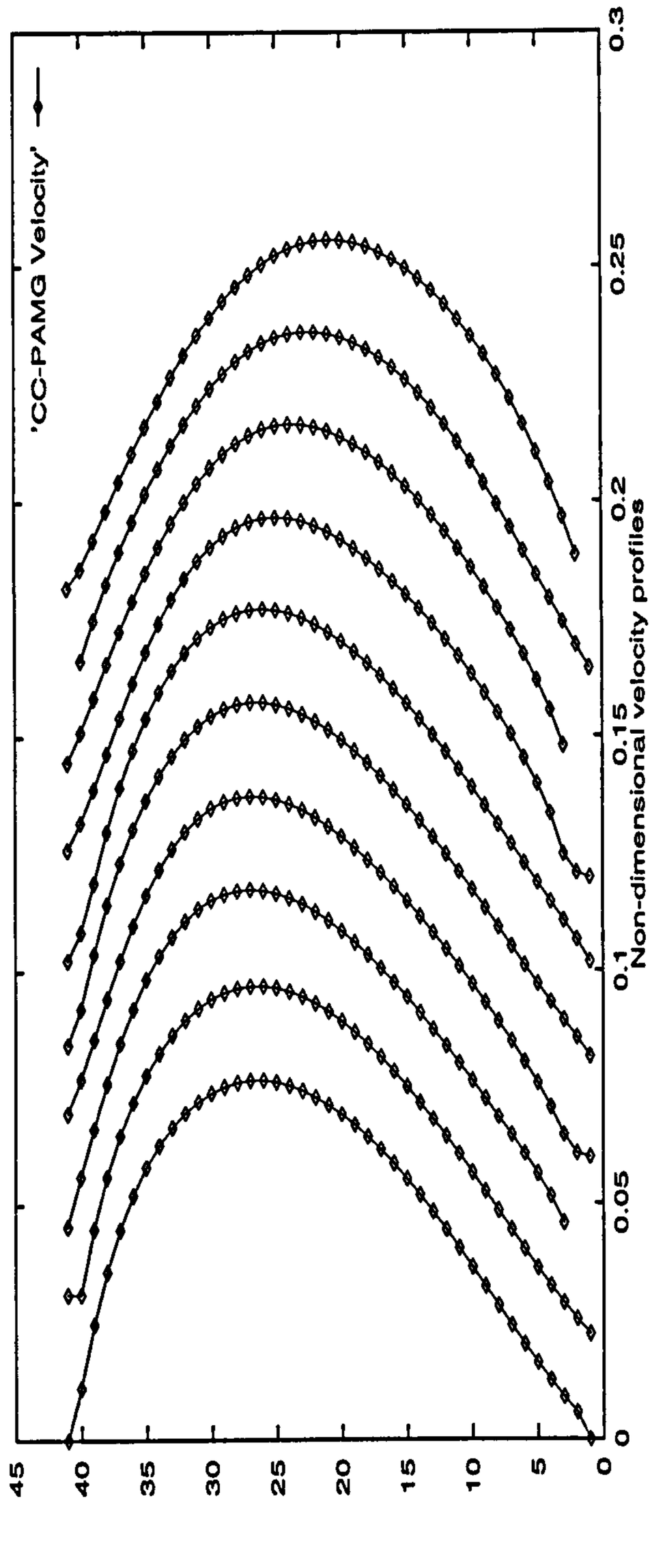


Figure 7.6: Curved channel flow problem - Velocity profiles along the bend zone for the CC-PAMG solution at Reynolds number = 100, uniform Inlet condition.(note polygon clipping has affected the ends of some of the profiles)

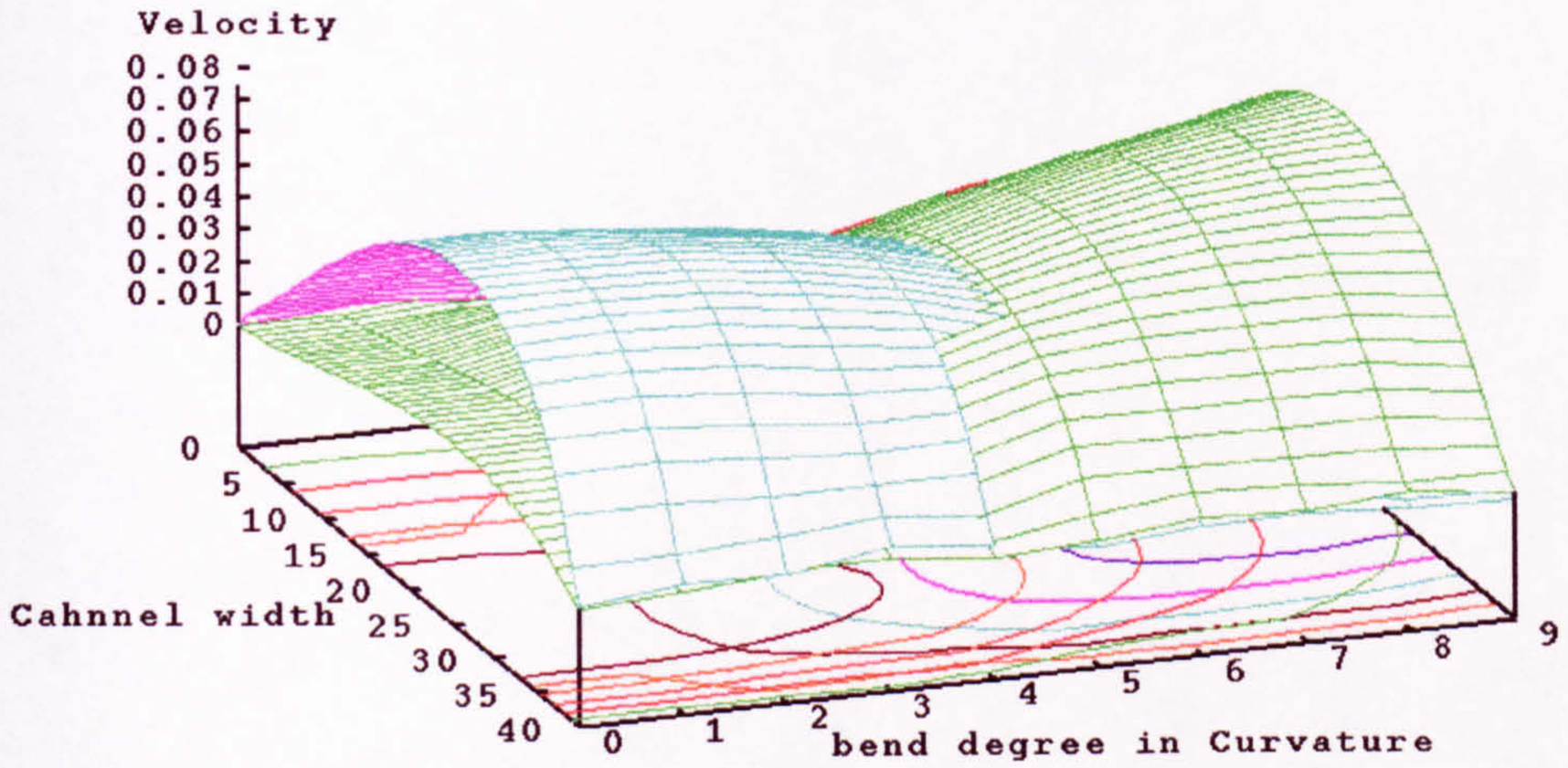


Figure 7.7: Curved channel flow problem - Velocity profiles along the bend zone while the inside velocity profiles are covered, notice the contours of the velocity components U and V are still can be discovered in the lower surface, Reynolds number = 100, uniform inlet condition. The channel width should be divided by 10, and the bend degree in curvature should be timed by 10 degrees

Four adaptive refinement levels of the velocity profiles are calculated at selected locations (Figure 7.8 to Figure 7.11). Comparing the four levels of the velocity profiles at selected position, it is shown that they match each other very well. The reason is that the grid size is small enough to deal with the curved boundaries, we cannot improve the accuracy by decreasing the calculation steps. The results are nearly the same except in the boundary layers, the difference are most pronounced near the boundaries. The refinement process improves the accuracy of the results near the walls. This clearly shows our novel treatment is accurate.

General velocity profiles calculated by the CC-PAMG algorithm produce satisfactory results as can be seen by comparing with results calculated by the commercial CFD package - CFX 4.2, incorporating a flat input. As we expected, the CC-PAMG solutions and the CFX 4.2 results coincide along the vertical line $x = 6.0$ (after bend), $x = 12.0$, and $x = 17.0$ (outlet); and the horizontal lines ($y = 0.005$ in the inlet). From the solution profiles obtained for these two schemes, the conclusions we obtained are:

Consider the fluid passing through a curvature, the pressure increases along the outside of the bend which forces the flow to change direction in the bend, and the maximum velocity in the core also shifts to the inside of the curved boundaries (see Figure 7.5 and Figure 7.9). The change in the pressure gradients at the bend readily appears. The velocity vector distribution shows that the velocity increases in the curved channel and the maximum velocity in the core keeps to the middle of the channel after the bend. Comparing these results, they are nearly the same in the inlet and after the bend (90^0 degree bend). However, there are some distinguished differences: the maximum velocity in the core shifts slightly to the inside of the bend for the CFX 4.2 results before entering the bend; it also shifts slightly to the outside in the output position. The results of CC-PAMG has no great differences. Although the maximum velocity in the core shifts to the inside of the curved zone, it always remains near the middle of the channel after traversed the curvature. Notice the 'near wall' profiles, the solutions of the CC-PAMG are smooth and the results of CFX 4.2 change suddenly which may cause the 'near wall' boundary errors. there are two main

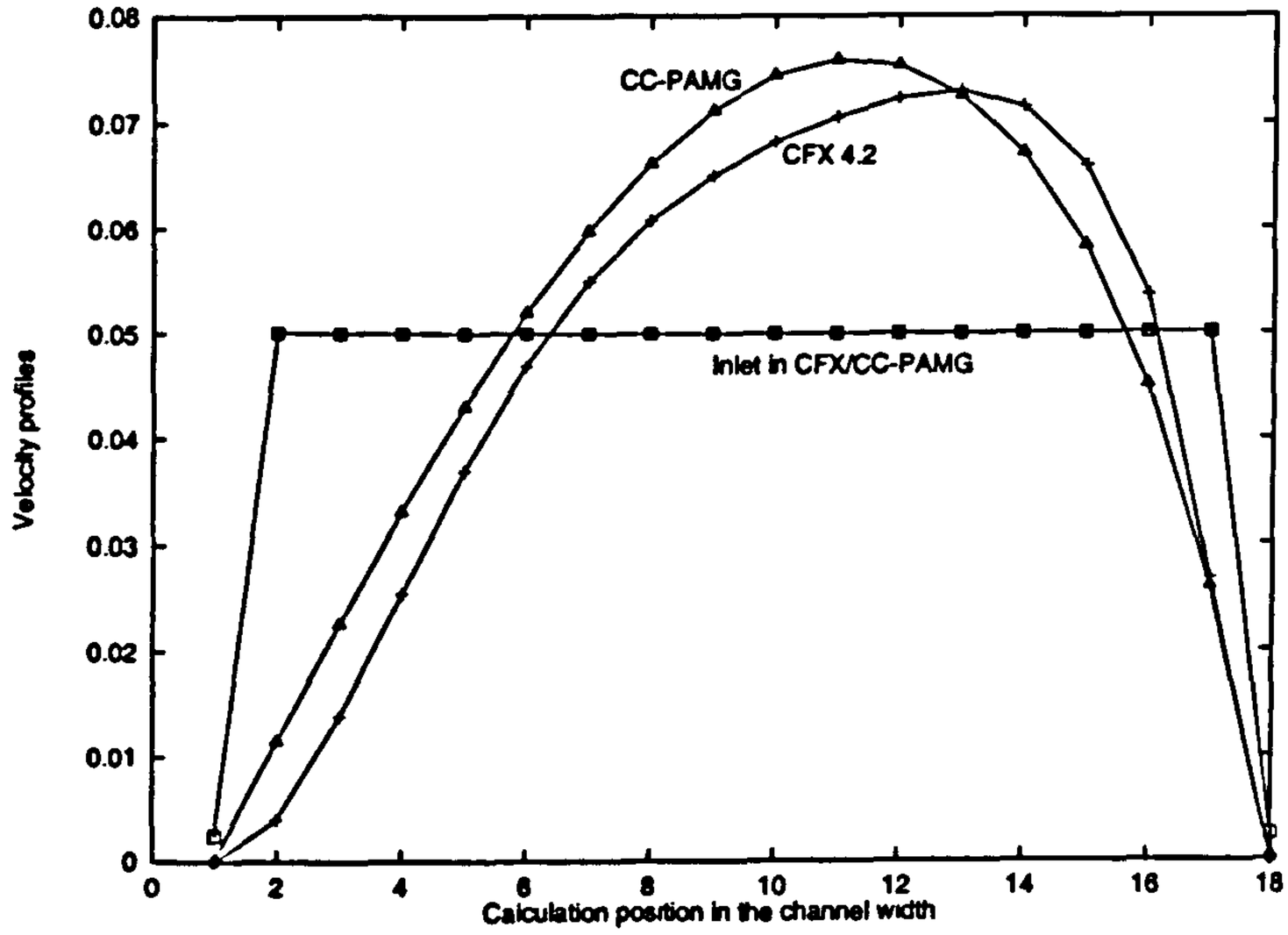


Figure 7.8: Curved channel flow problem - Velocity profiles at inlet and before bend. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

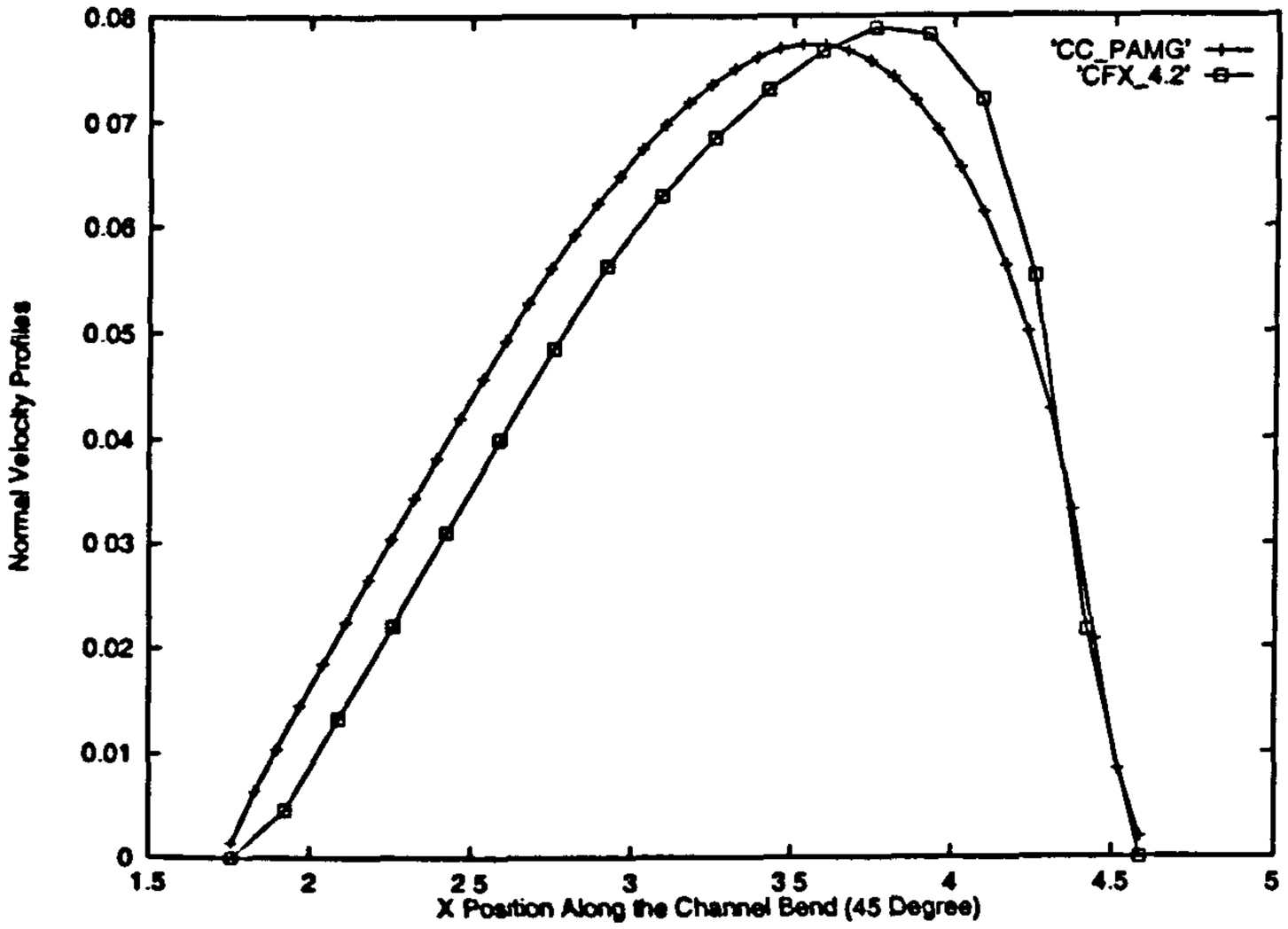


Figure 7.9: Curved channel flow problem - Velocity profiles at position of 45 degree bend. Comparison of the CC-PAMG and CFX 4.2 Solutions at Reynolds number = 100

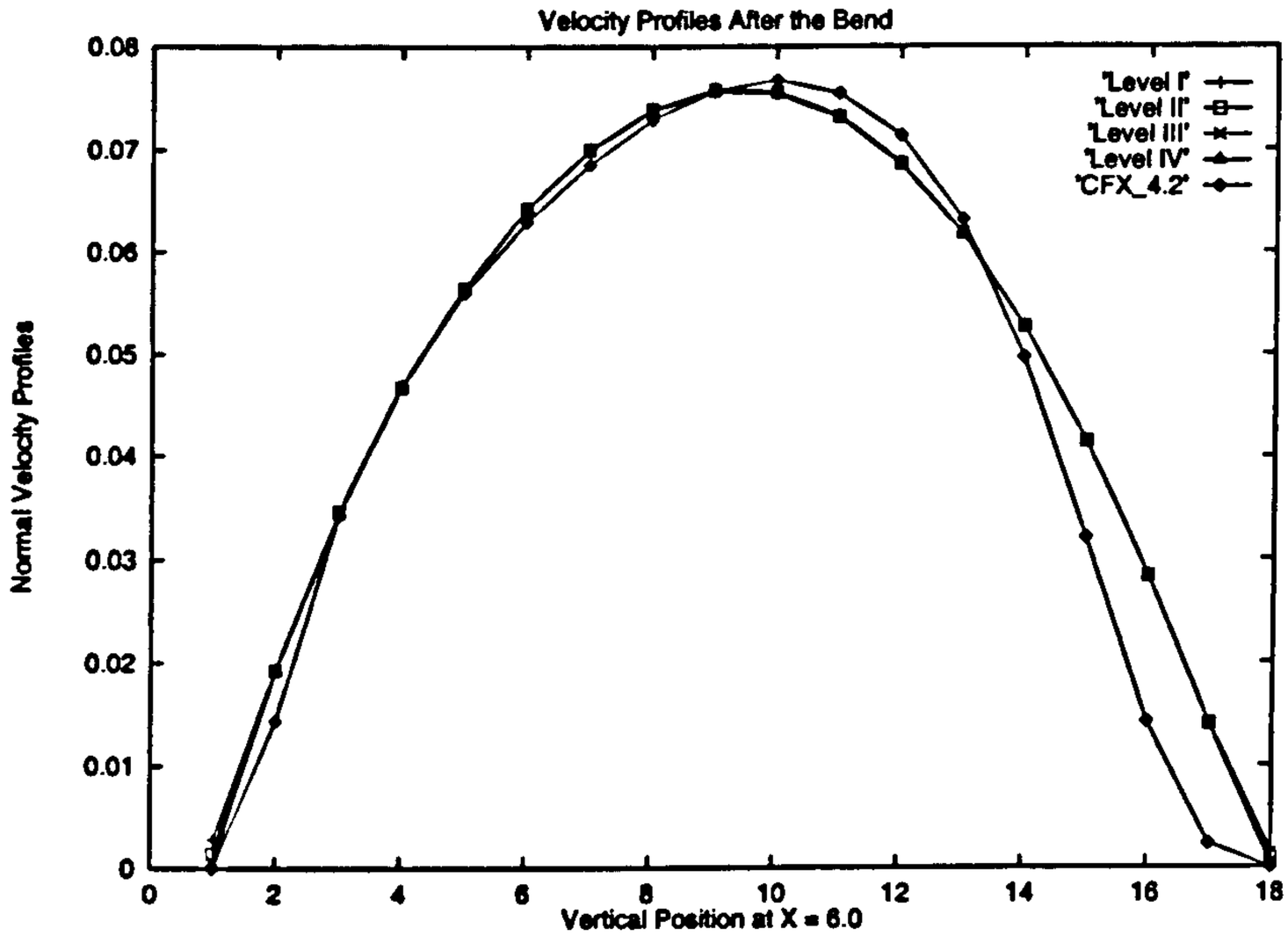


Figure 7.10: Curved channel flow problem - Velocity profiles after bend (90 degree bend). Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

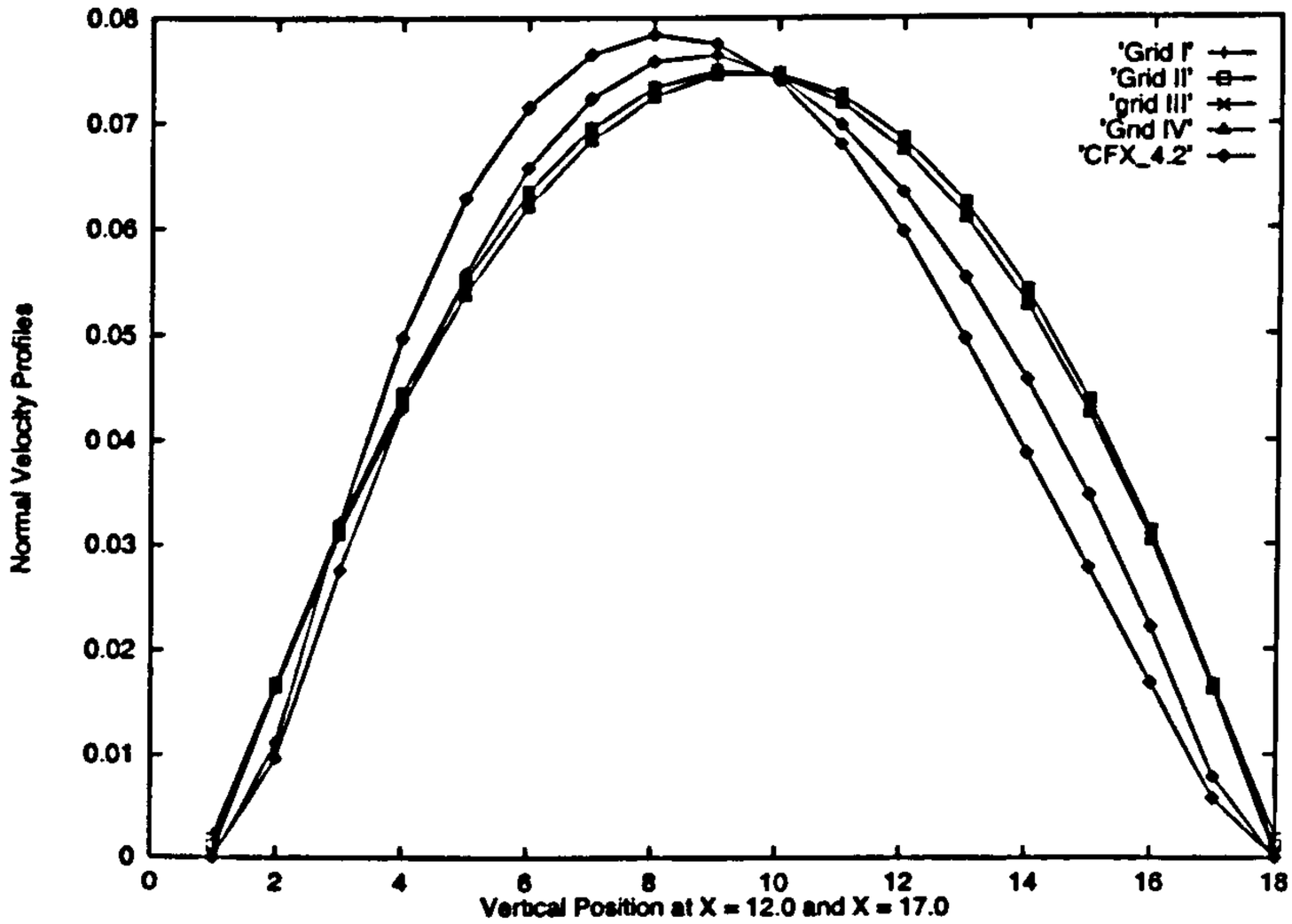


Figure 7.11: Curved channel flow problem - Velocity profiles along the lines x =12.0 and X = 17.0. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

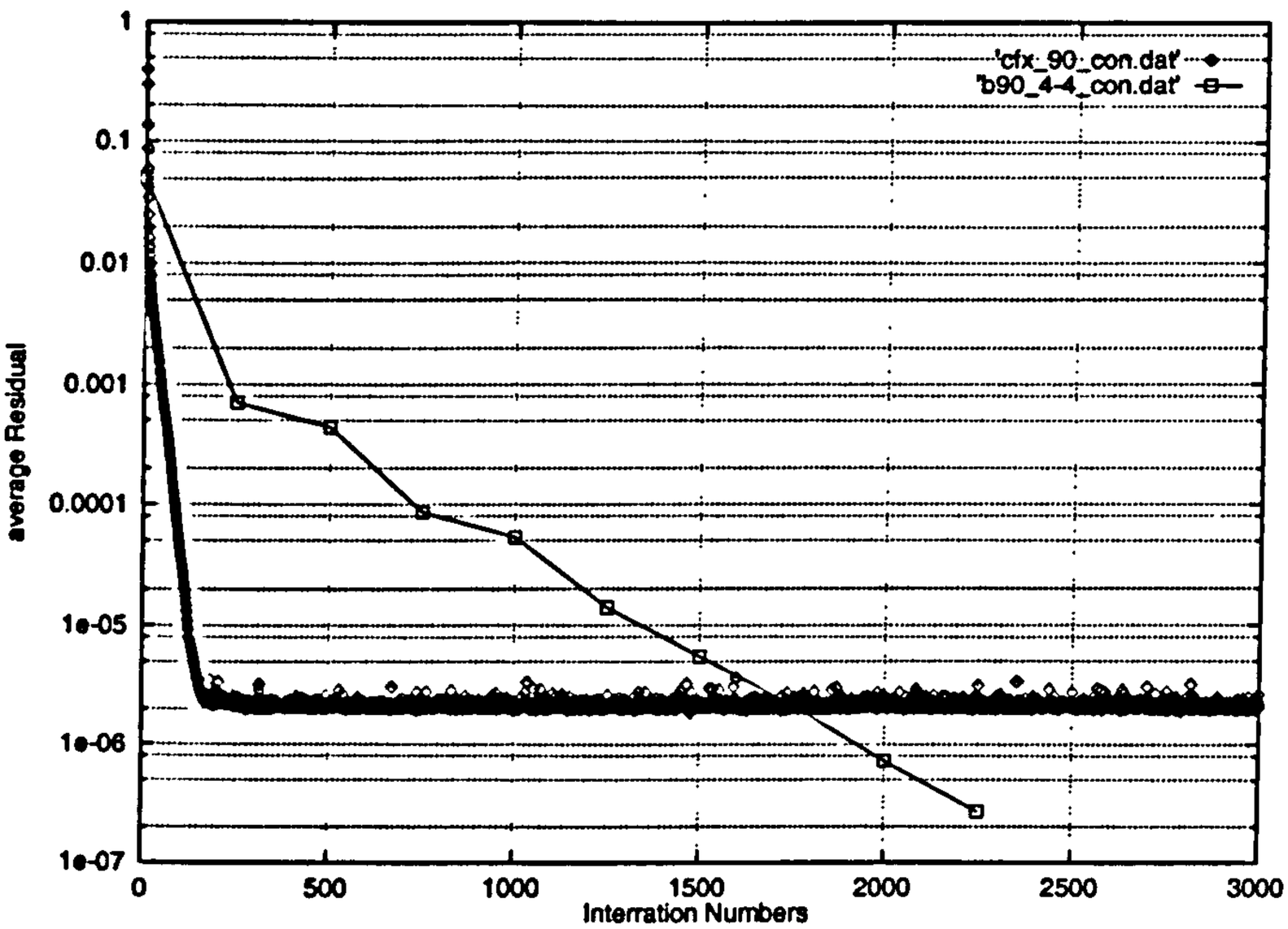


Figure 7.12: Curved channel flow problem - Comparison of the convergence histories of CC-PAMG (F(4,4) Cycles) and CFX 4.2 at Reynolds number = 100

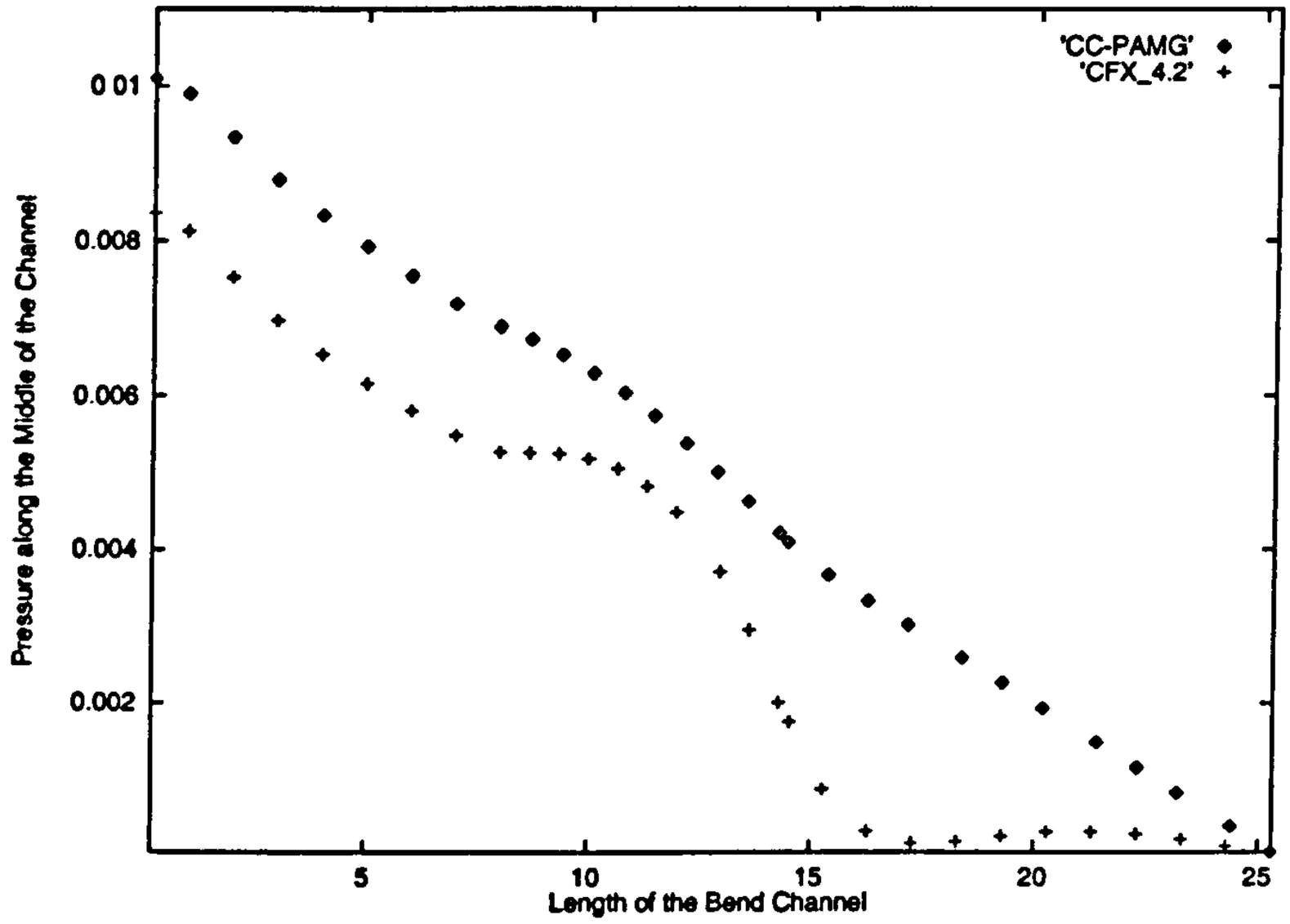


Figure 7.13: Curved channel flow problem - Comparison of the pressure drop ratio along th middle of the channel with CC-PAMG (F(4,4) cycles) and CFX 4.2 at Reynolds number = 100

reasons to effect the CFx 4.2 results: first, the CFX4.2 cannot reach the same residual tolerance as in CC-PAMG (10^{-6}), it is introduced some residuals; the second reason is this two algorithms using different grid size. In the CFX 4.2, a slightly big cell size is used ($\delta x = \Delta y = 0.25$ unit). In CC-PAMG, eight patches are used to occupy the channel width, so the cell size in CC-PAMG is smaller (0.125 unit). according to the Richardson Extrapolation Theory (Section 4.5.3), the CC-PAMG should have 4 times smaller residuals than the CFX 4.2. Comparing the results of different level in non-aligned channel flow (figure 6.43), it is improved that the CFX 4.2 will have the same results when the cell size reduced.

Numerical results of the pressure drop

In the theoretical solution for a channel, the maximum velocity in the middle of the channel is a function of the pressure drop, channel width, and viscosity. Due to the irregular boundaries and curved channel flow, the calculated pressure in the middle of the channel is no longer along one direction. Figure 7.13 shows two different pressure profiles (CC-PAMG and CFX 4.2) along the middle of the channel. The pressure drops along the middle of the channel are obtained by dividing the channel with 30 points, and calculating the local pressure drops, then adding them together and averaged them to get the whole pressure drop along the middle of the channel. The pressure drops are shown as follows:

Pressure Drop Ratio		
Channel position	CC-PAMG	CFX 4.2
before bend	-4.515 $\times 10^{-04}$	-4.402 $\times 10^{-04}$
bend	nearly linear	non-linear
after bend	-3.415 $\times 10^{-04}$	-3.15 $\times 10^{-05}$

Table 7.1: Pressure drop proportions along the middle of the channel

Comparing these results for the pressure drop ratios, the pressure drops have

decreased in a nearly constant factor in CC-PAMG results, although they have a significant difference comparing CFX 4.2 solutions. Before the bend, the pressure drop ratios are nearly the same. In the curved zone, the pressure drop rates are disturbed in the curved lines by the bend. The worst result take place after the bend in the CFX 4.2 results. The reason maybe due to the fact that the two schemes have different convergence tolerances. The CFX 4.2 results cannot approach the given tolerance 10^{-6} which the smaller pressure drop affects the results (see Figure 7.12).

The Performance of the two Solvers

We have established the accuracy of the bent channel solutions provided by CC-PAMG. We will now focus on the efficiency of the solver and convergence factors observed with the Cartesian cut-cell algorithm. In numerical convergence is usually monitored by the reduction of the residual, measured using a suitable vector norm. Solution is not relevant.

In this section we therefore concentrate on the convergence factors obtained with CC-PAMG in the curved channel flow. The ideal case would be that the convergence factors are grid-independent, ensuring that the method is optimal order-wise. It should be noted that the convergence factors which to a large extent, we have achieved are grid independent, in significant in parts of the convergence histories.

The convergence histories are summarised in Figure 7.12 for uniform inlet flow. They clearly demonstrate that CC-PAMG out-performs a good segregated solver. It is shown that the CC-PAMG solver has a good convergence rate which saves a lot of computational time. The CFX 4.2 calculations were performed with the default under-relaxation factors. For many cases, a greater amount of under-relaxation is necessary to obtain a converged solution. The performance of the commercial CFX code is shown in Figure 7.12. Figures 7.14 to Figure 7.21 show the convergence histories of the CC-PAMG schemes for parabolic inlet flows. The mesh refinement improves the solutions near the walls through each level of refinement, however, the

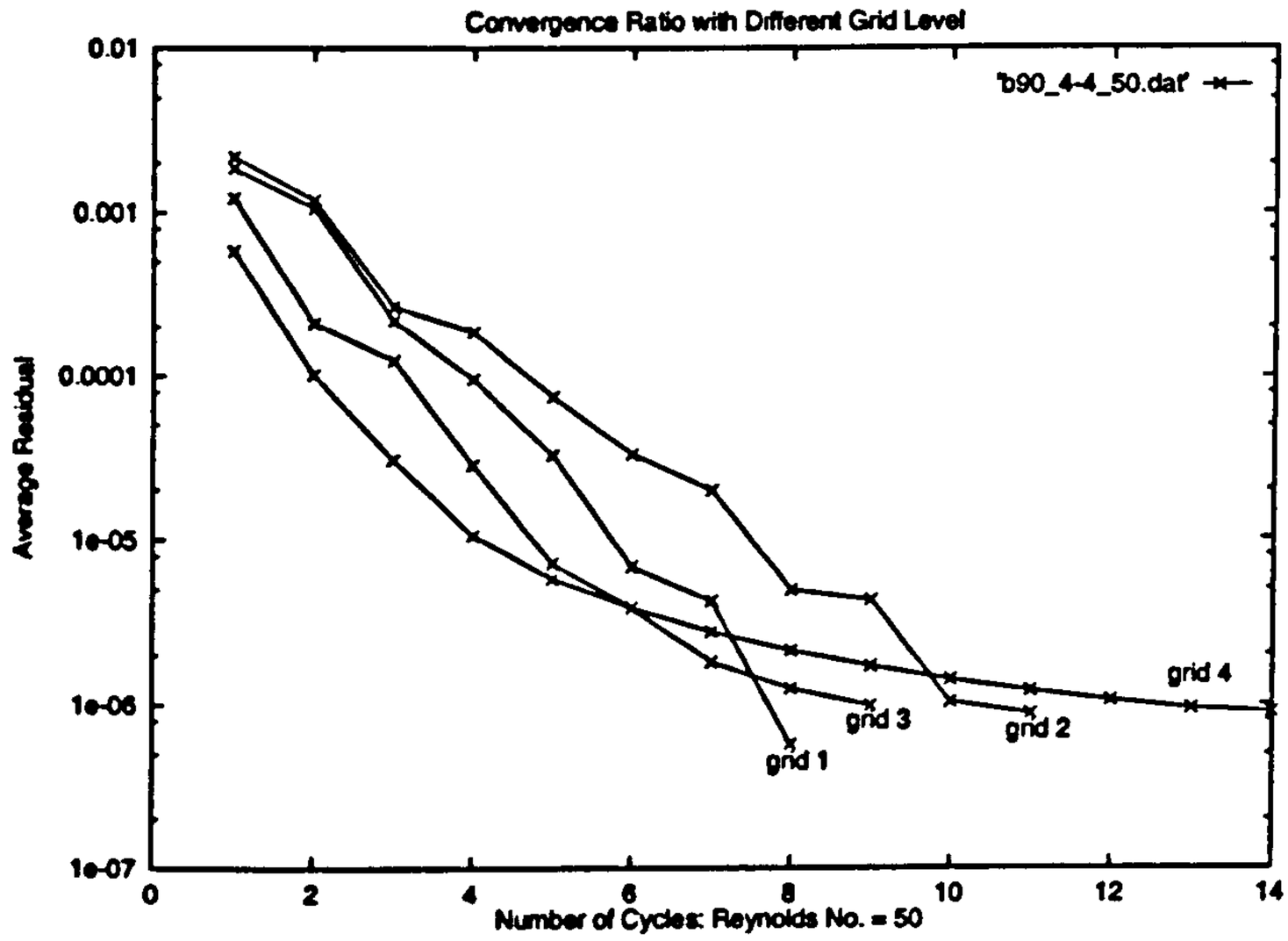


Figure 7.14: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(4,4) cycles with Reynolds number = 50

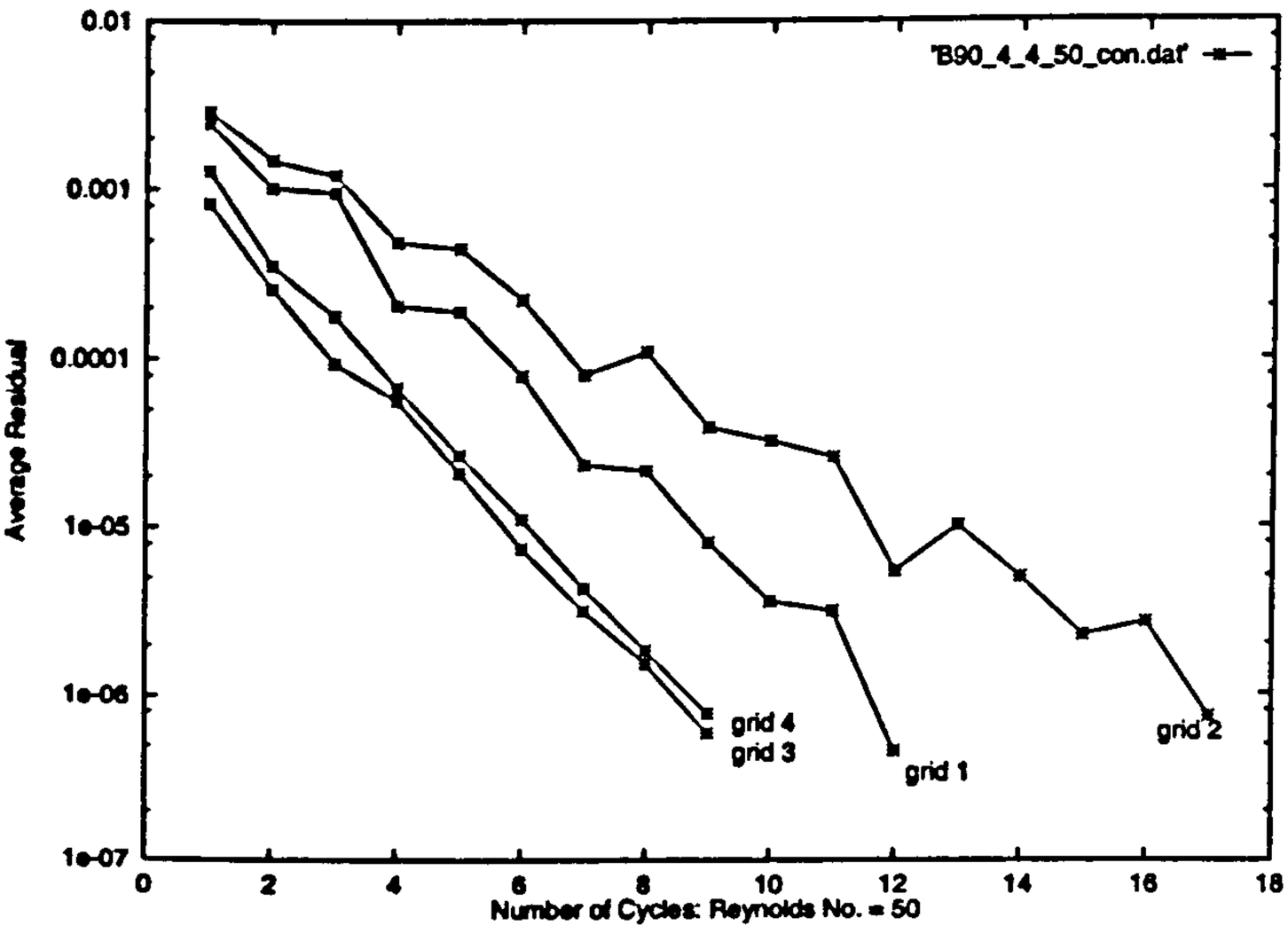


Figure 7.15: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(4,4) cycles with Reynolds number = 50 after modification

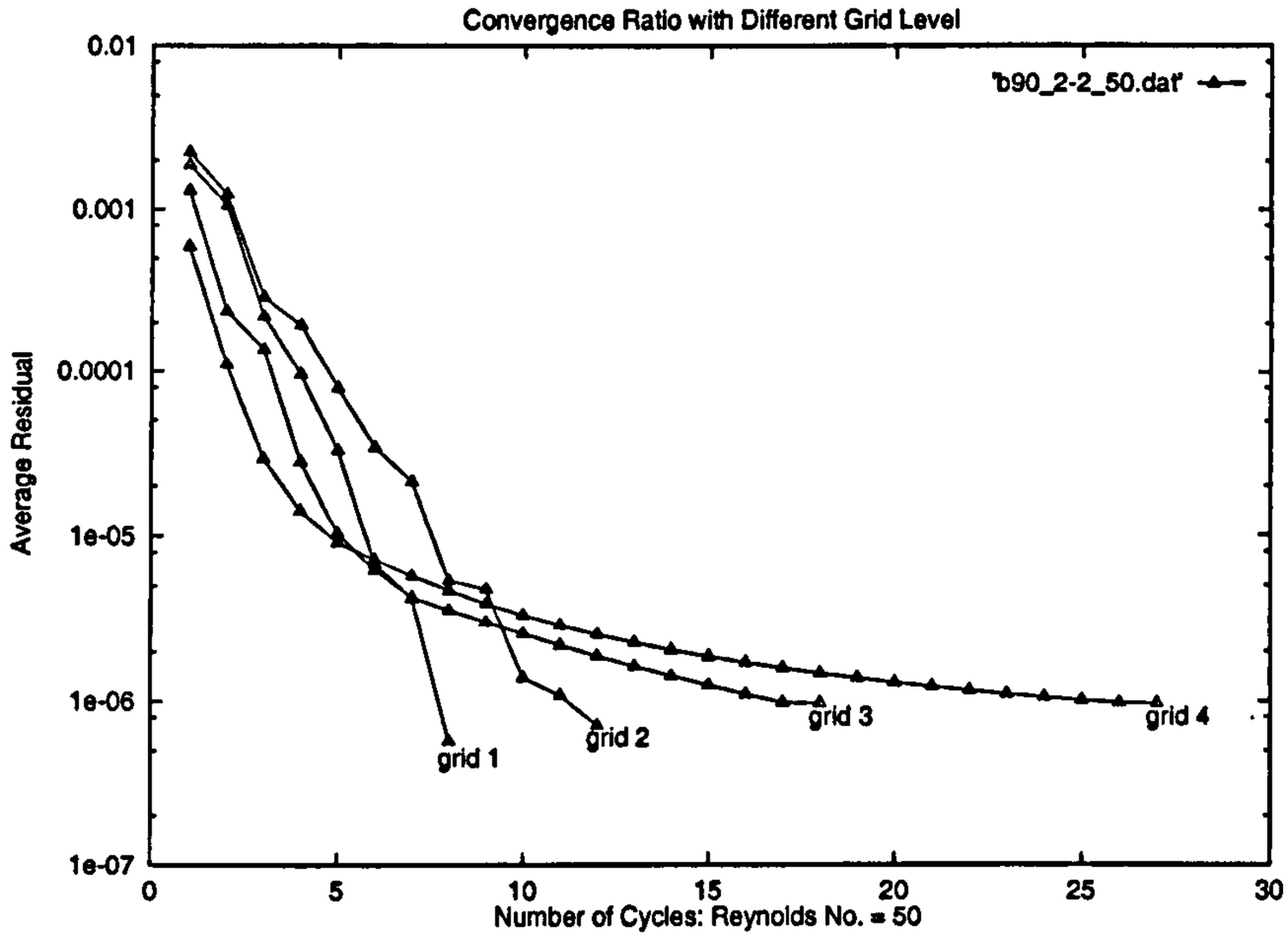


Figure 7.16: Curved channel flow problem - Convergence of CC-PAMG Solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 50

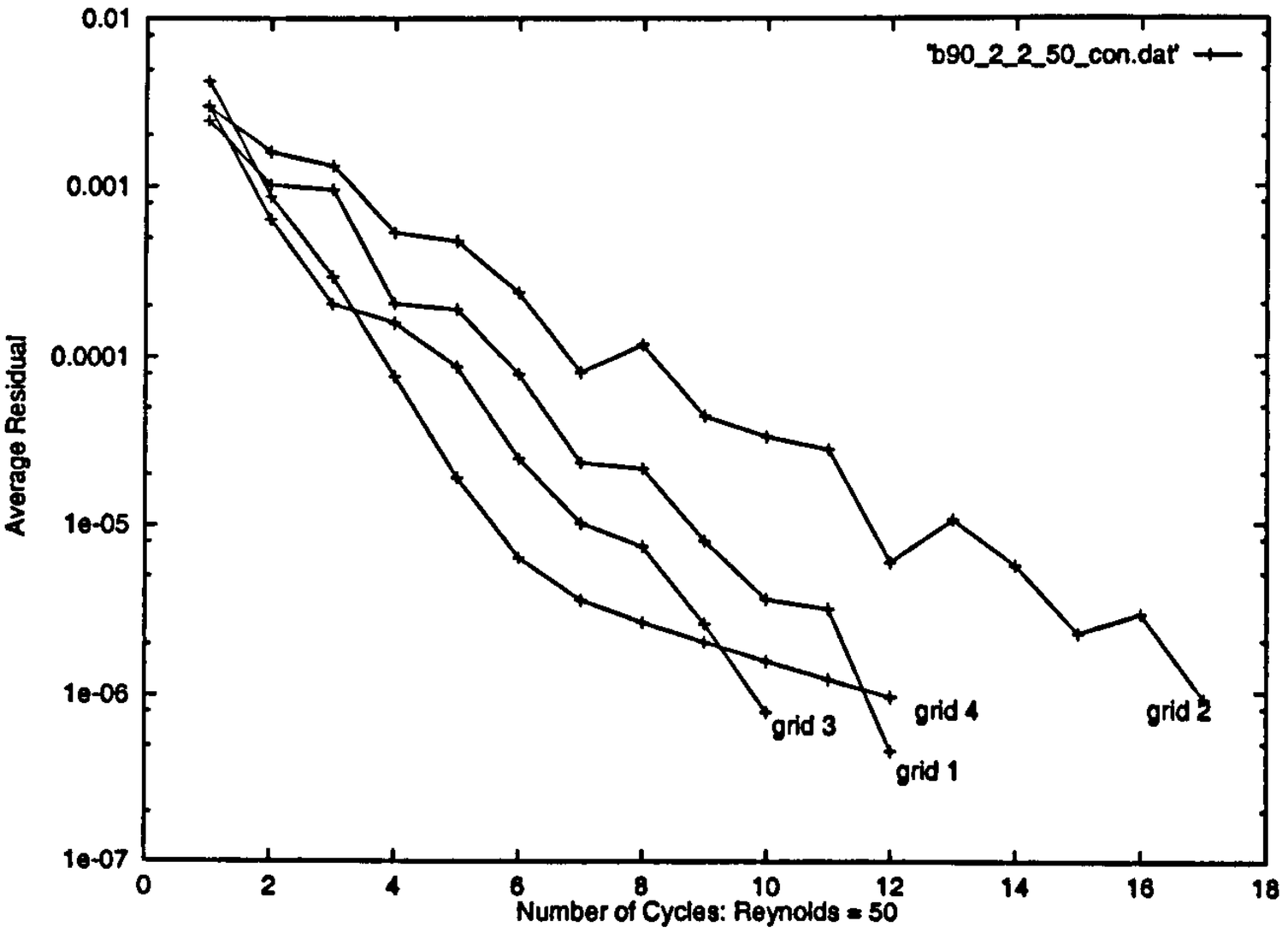


Figure 7.17: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 50 after modification

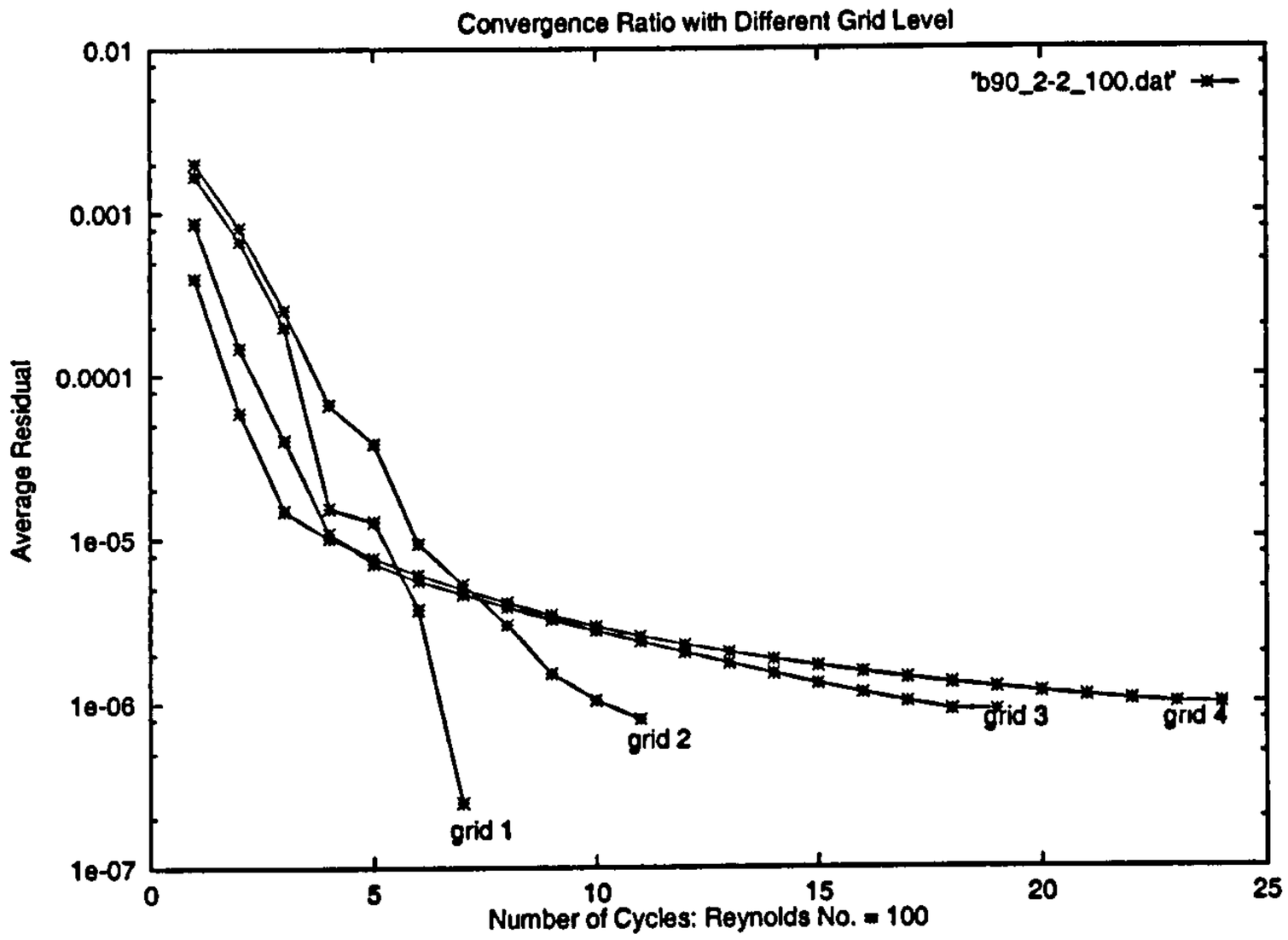


Figure 7.18: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 100

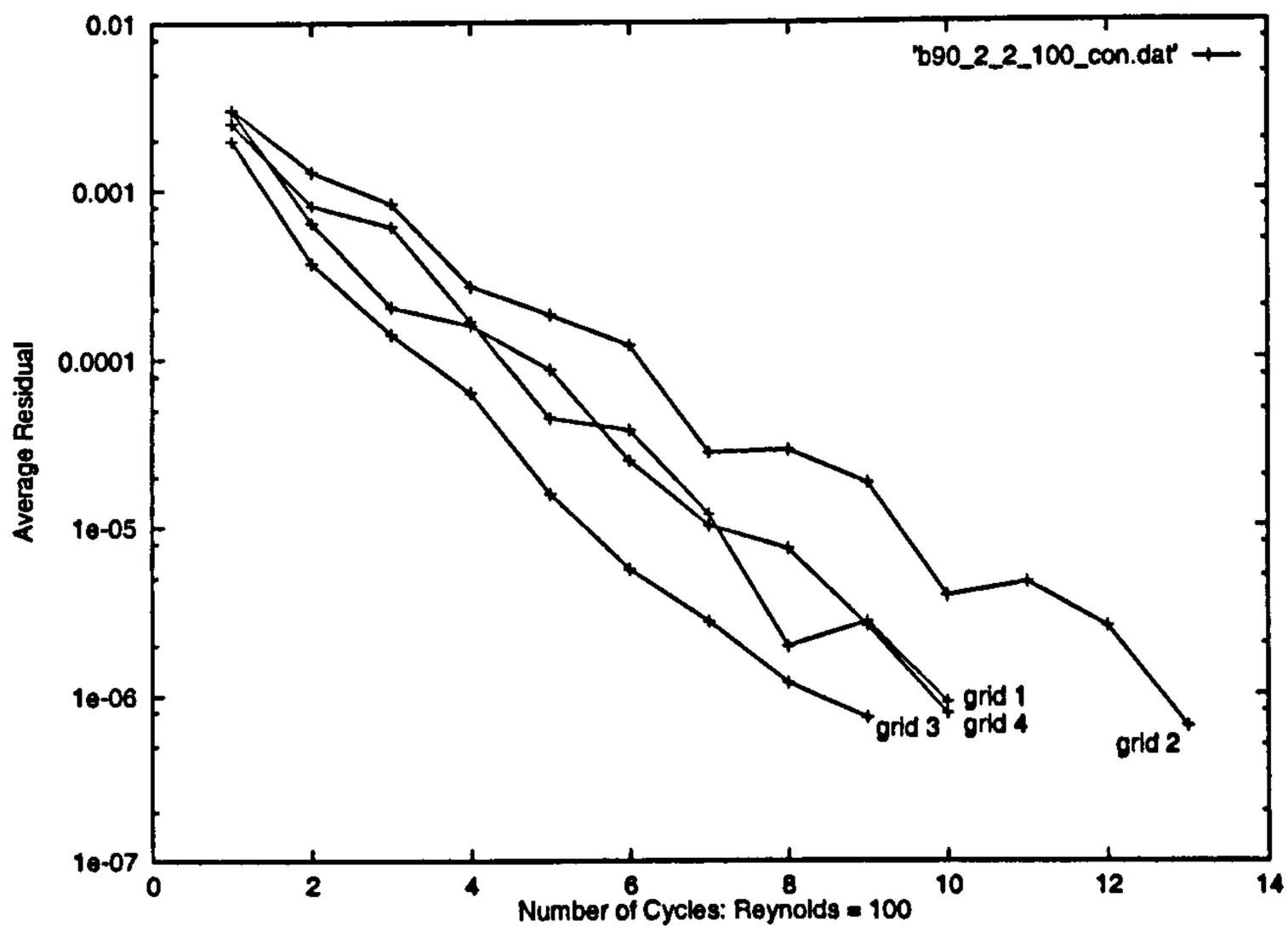


Figure 7.19: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 100 after modification

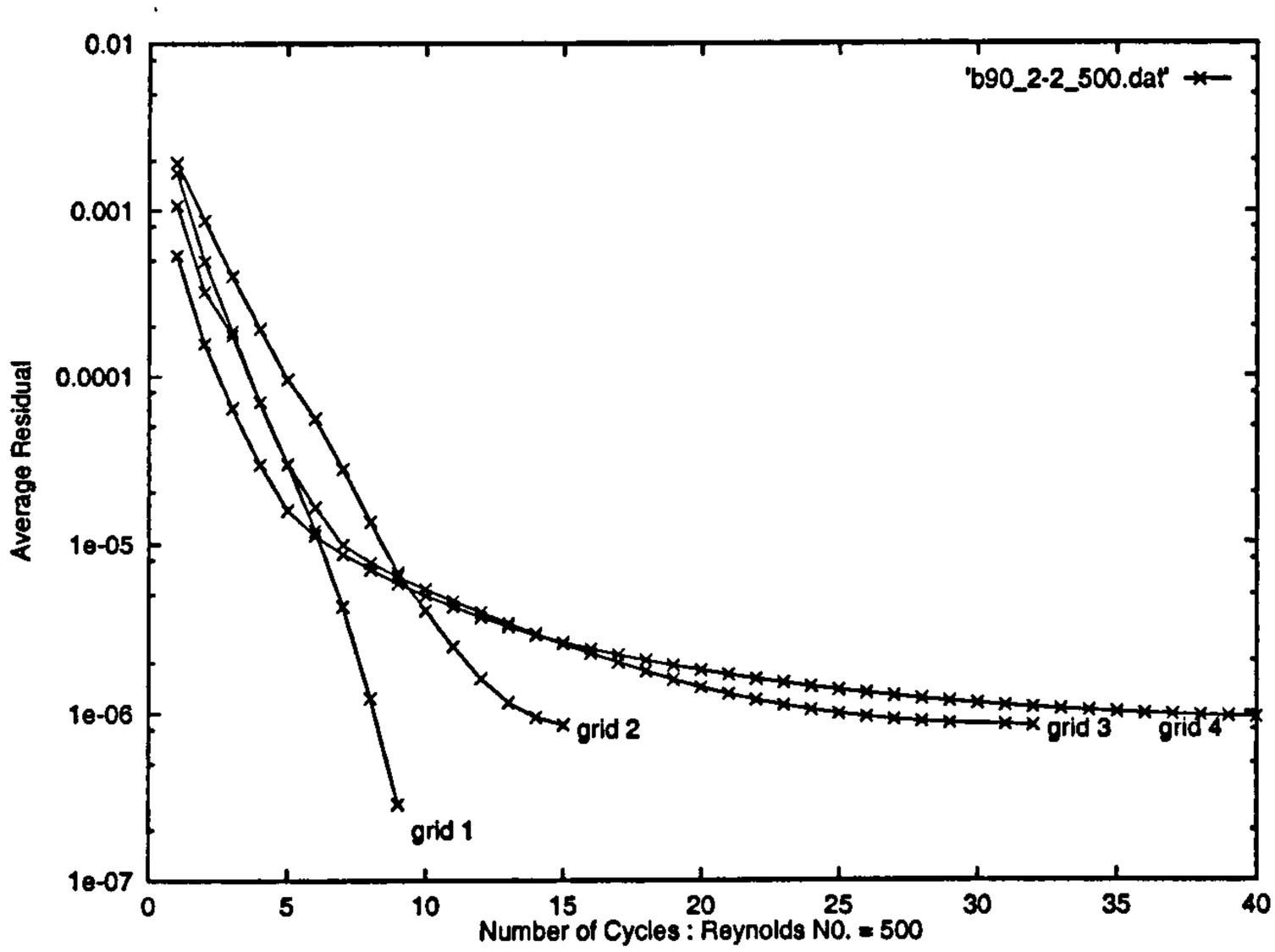


Figure 7.20: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 500

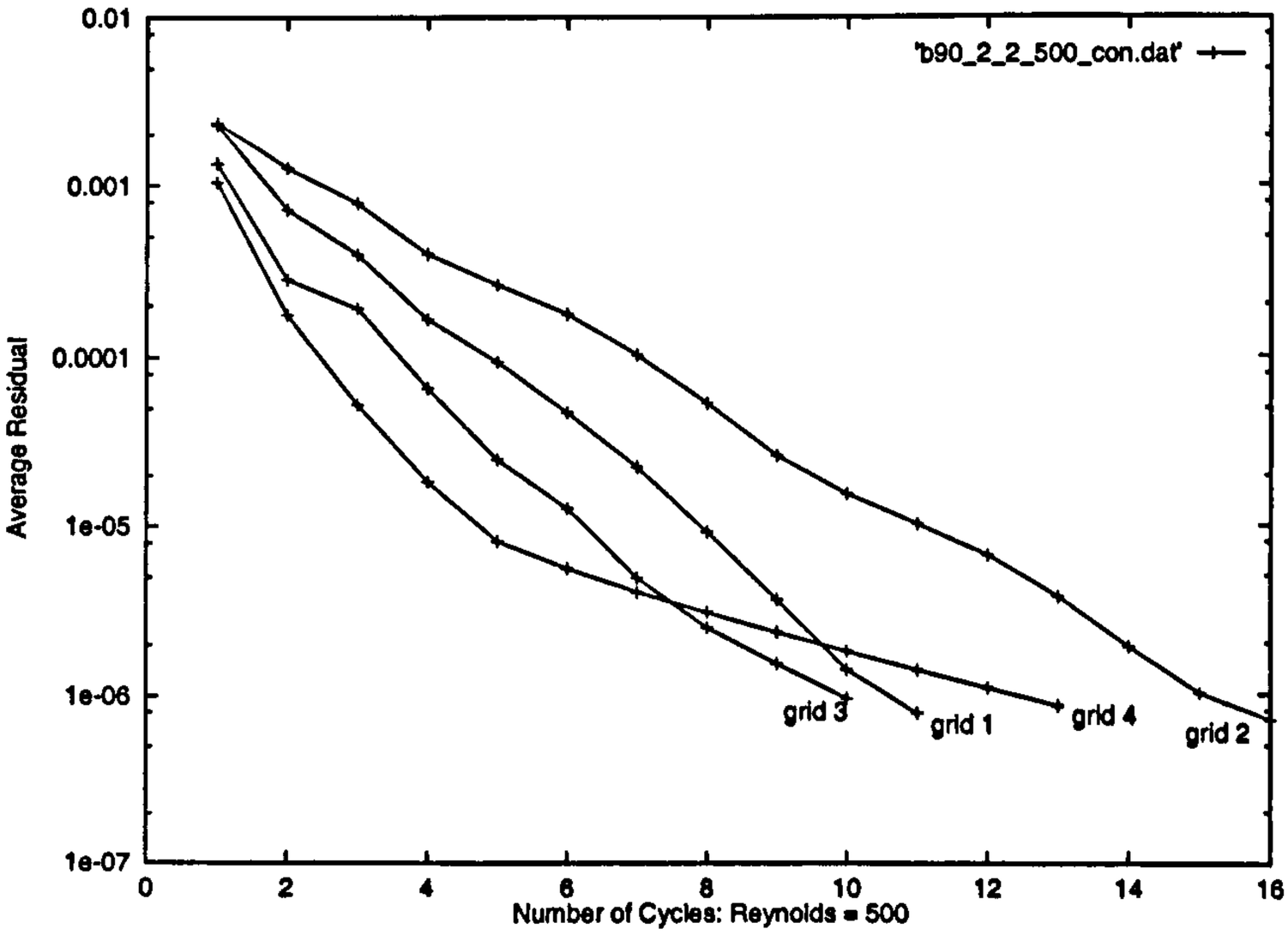


Figure 7.21: Curved channel flow problem - Convergence of CC-PAMG solutions for uniform grid computation from grid 1 to grid 4, F(2,2) cycles with Reynolds number = 500 after modification

Rey. No.	Grid 2	Grid 3	Grid 4
50	12	18	27
100	11	19	24
500	15	32	40

Notes: F(2,2) cycles, Power=2.0, tol.=10^{-6}

Table 7.2: Comparison of the convergence reduction at different Reynolds numbers

CC-PAMG algorithm converged quicker at higher F-cycle numbers. For example, at higher refinement grids level 4, the F(2,2) cycles needs around 27 cycles to reach the tolerance limit, but the F(4,4) cycles only needs 14 cycles to reach the tolerance at the same value of the Reynolds number. Of cause, F(4,4) is twice the work of F(2,2), so this is expected. After the boundary residual treatment, the convergence ratio changes dramatically (comparing these convergence figures and we will discuss the reason later).

Discussion

Adaptation allows the grid not only to resolve properly highly curved regions of the bodies, but also to resolve the disparate length scales in the flow. The existence of small cut cells is a problems with Cartesian grids. The small cut cells can lead to inaccuracies in the flow by becoming decoupled from the rest of the flow. From the convergence histories, it is found that the convergence rate becomes much worse after they reach 10^{-5} for 3 and 4 levels. The effect of the Reynolds number for the convergence reduction has also been investigated. Table 6.2 gives the different convergence reduction results with different Reynolds number. In this particular case, the F(2,2) cycles requires only 24 cycles to reach 10^{-6} at low Reynolds number (Re = 100, say). At higher Reynolds number (Re = 500), the same cycles must have 40 cycles to approach the same tolerance (see Figure 7.18 and Figure 7.20).

With careful investigation, we found that the biggest error occurred at the bend boundaries, especially at the entry and exit to the bend. For example, the residual at the beginning of the curve on the inside bend is reduced sufficiently on grids one and two. However, the residual at level 3 and level 4 cannot be reduced properly at low smooth process. At this position the pressures are relatively small and the maximum velocity profiles are shifted to the inside of the bend, the velocity gradient is relatively large adjacent to the inside wall. The residuals of the continuity equations cannot be reduced at low smooth process, it needs more smooth processing to reduce the residuals further. We have observed that velocity corrections are not changed so much with different grid levels near the inside bend walls, the residuals will depend on the mesh spacing. If the size of the steps is smaller, then the residual is bigger.

Modification

If the grid is located on the boundary, the two velocity components are cut out in the staggered grid. The residual in the continuity equation is very difficult to reduce in this situation, which will affect the convergence rate for the computational domain. Consider the simple case shown in Figure, when the wall is bent through a angle α, the flow upstream will not be influenced by the wall change, but downstream the fluid must flow in the new direction parallel to the wall and will have its pressure and velocity changed slightly (Figure 7.22). So the velocity near the bent wall will bigger than the neighbour cells in the uncurved wall, it causes the bigger residual and difficult to decrease the residuals.

To overcome these difficulties, we ignore the residual at first in the cut-cell, however the inside neighbour cells have been disturbed by this treatment, the convergence rate is even worse. Finally, we found that the cut-cell residual of the continuity equation should be used as a reduced residual. The reduced residual is divided by the square of the grid level, and the convergence rate changes dramatically (compare Figure 7.20 and Figure 7.21) after the following modification:

$$resu_c = \left(\frac{\Delta u_{i,j} - \Delta u_{i-1,j}}{\Delta x} + \frac{\Delta v_{i,j} - \Delta v_{i,j-1}}{\Delta y}\right)\frac{1}{l^2}$$

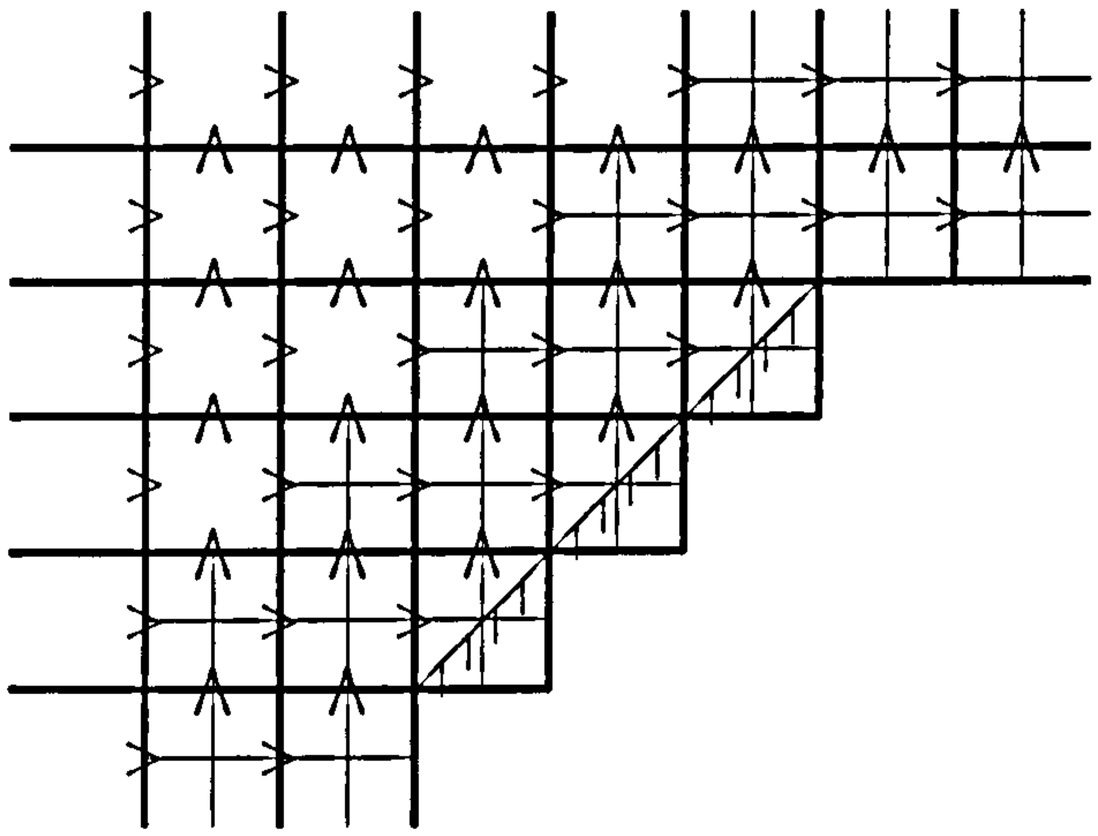


Figure 7.22: Cut-cell boundary for the integration of the continuity equation

where l indicates the level of the grid refinement.

Local Relaxation Algorithm

When we investigate the residuals in the calculation domain, it is found that the residual can not be reduced at the same rate as we expected, some parts of the residuals reduced quickly, and on the other hand, especially in the curved boundaries and leading edge positions, the residuals reduced slowly. some critical positions even not reduced in higher level after the prolongation and restriction process, A new method should be developed to overcome this problems. This is so called local relaxation method.

The local relaxation procedure considered in this thesis results in a composition grid with locally relaxation regions. Adaptive, local grid relaxation is combined with a multigrid solution technique. The average residuals of the calculation domain is computed after each relaxation trial. comparing the individual residual with the previous average residual, Local relaxation method is used to reduce the residuals in the particular patches if the residual is big than the given threshold (average residual).

```
        If Initial ER_Avg = 10^20
        Do 20 J = 1, Numpts
            Call Relax( U_s, V_s, P_s, E_2, ER_Local)
888     Continue
        If (ER_Local ≥ ER_Avg) then
            Call Relax( U_s, V_s, P_s, E_2, ER_local)
            Go To 888
        End if
        ER_Total = ER_Total + ER_Local
20      Continue
        ER_Avg = ER_Total/Numpts
```

Figure 7.23: Pseudo-code procedure to set up local relaxation subroutine

The pseudo-code shown in Figure 7.23 will co-ordinate the complete local relaxation algorithm.

The basic structure is as follows. An initial relaxation is performed for every patch. At the same time a mean residual is calculated for all patches at a given level. (This is accomplished by initialising the mean to ∞). A second pass is then performed where each patch is visited once and patches with a large error are relaxed until its errors is less than the mean.

In terms of the number of relaxations as well as the time spent in in relaxation, the conclusion seems to be that the local relaxation method reduced the computational time significances. In Table 7.3, the time spent and average relaxation times are reduced to one-third compared with the original CC-PAMG. Although the reduction is only around three-fifth of the number of calls to the relax procedure, but we saved

Table 7.3: Time spending and relaxation calculate for the Non-aligned channel flow: Original CC-PAMG Calculations and CC-PAMG Local relax Calculations at Reynolds number =100

		CC-PAMG	Local Relaxation Results		
No. of Relax Sweep in Grid 1		200	100	50	25
Grid 2	No. Relax	13,080	4,894	4,156	6,161
	No of cycle	8	4	3	4
	Time(sec.)	2.7	1.0	0.8	1.2
	Ave. relax	545	204	173	156
Grid 3	No. Relax	32,694	9,001	11,130	24,730
	No of cycle	22	7	9	15
	Time(sec.)	6.6	1.7	2.1	4.5
	Ave. relax	389.2	107.2	132.5	294.4
Grid 4	No. Relax	483,324	342,971	297,935	311,584
	No of cycle	18	10	9	12
	Time(sec.)	92.4	62.5	51.4	57.3
	Ave. relax	433.1	307.3	267.0	279.2

Notes: CC-PAMG F(4,4) and Local Relaxation F(2,2)
cycles with Reynolds No.= 100, tolerance = 10^{-6}

much more computational time (92.4:51.4 sec.) at highest level.

For the more complex geometry model: curved channel flow (Table 7.4), the solutions are nearly the same at the coarsest grid, the average relax times are not much difference (2800:2636.5). However, at the higher levels, it changes dramatically as in the non-aligned channel flow. Many computational times are saved in level 4 although the two numbers of average relaxation are only one-sixth difference. The reason is that the local relaxation need less prolongation and restriction process in the multigrid algorithm.

Table 7.4: Time spending and relaxation calculate for curved channel: Original CC-PAMG Calculations and CC-PAMG local relax Calculations at Reynolds number =100

		CC-PAMG	CC-PAMG Local Relax
No of Relax Sweep in Grid 1		350	325
Grid 1	No. Relax	1,148,000	1,080,952
	No of cycle	8	5
	Time(sec.)	204.6	183.5
	Ave. relax	2,800	2,636.5
Grid 2	No. Relax	2,961,616	2,035,635
	No of cycle	10	9
	Time(sec.)	548.0	346.4
	Ave. relax	1,451.1	997.4
Grid 3	No. Relax	7,159,460	4,167,432
	No of cycle	6	7
	Time(sec.)	837.7	487.6
	Ave. relax	1,351.6	708.4
Grid 4	No. Relax	22,482,970	18,329,294
	No of cycle	6	7
	Time(sec.)	4,279.4	3,169.1
	Ave. relax	651.0	531.0

Notes: CC-PAMG F(90,4) and Local Relaxation F(10,4) cycles with Reynolds No.= 100, tolerance = 10^{-6}

It may be concluded that the local relaxation method improved the convergence rate at adaptive multigrid algorithm with the Cartesian cut-cell in the boundaries. It is clear that the time spends in the iteration process reduced quickly and less time needs on the prolongation and restriction processes. The computational cost of cause is also reduced dramatically.

7.2.2 Numerical Solution for Bent Channel (180^0) Flow

In order to test the ability to deal with a more complex geometry using the CC-PAMG, the laminar flow in a semi-circular curved channel has also been investigated. The Reynolds number is taken to be 100. The size of the channel is 4.0 wide as like the previous bent channel but this is bend through 180^0 and the equivalent length of the channel is 34.4. The flat inlet flow is used to the same computing domain with the commercial package CFX 4.2 and CC-PAMG algorithm. No-slip boundary conditions are applied on the semi-circle curvature and other boundaries as before.

The streamlines and pressure distribution profiles of the CC-PAMG solution are shown in Figure 7.24. In order to investigate the more complex geometry, a small grid size is used to approximate the calculated domain. In this semi-circular bent channel, the velocity vectors change gradually from the vertical direction to the horizontal direction then to the negative vertical direction (if possible refer to CC-PAMG Figure 7.27, CFX 4.2 solutions); and the streamlines smoothly passed the semi-circular zone (Figure 7.24, CC-PAMG solutions). A diagram of the adapted grid for this semi-circular bent channel at the first mesh level is shown in Figure 7.25. Figure 7.28 to Figure 7.32 show the improvement of the solutions due to mesh refinement, at given locations. Both schemes (CC-PAMG and CFX 4.2) are compared to each other at a selected series of locations in Figure 7.28 to Figure 7.32.

As in the previous section, the boundary geometries in the curved zone are displayed as stepped boundaries. Note that in Figure 7.24 to Figure 7.25, they have stepped boundaries in the semi-circular curved zones. Although the curved boundaries are approximated by straight lines not the curved lines, which the curved boundaries are approached by series of linear lines, the calculated solutions along both un-smooth boundaries have little disturbance by the linear lines approaching. However, the velocity near the wall is slow, and the un-smooth curved channel wall has little effect on the velocity profiles. The velocity profiles close to the wall are very small and nearly smooth. Figure 7.26 shows the velocity profiles along the semi-circle region change gradually from the vertical direction (0^0 offset) to the horizontal (at 90^0

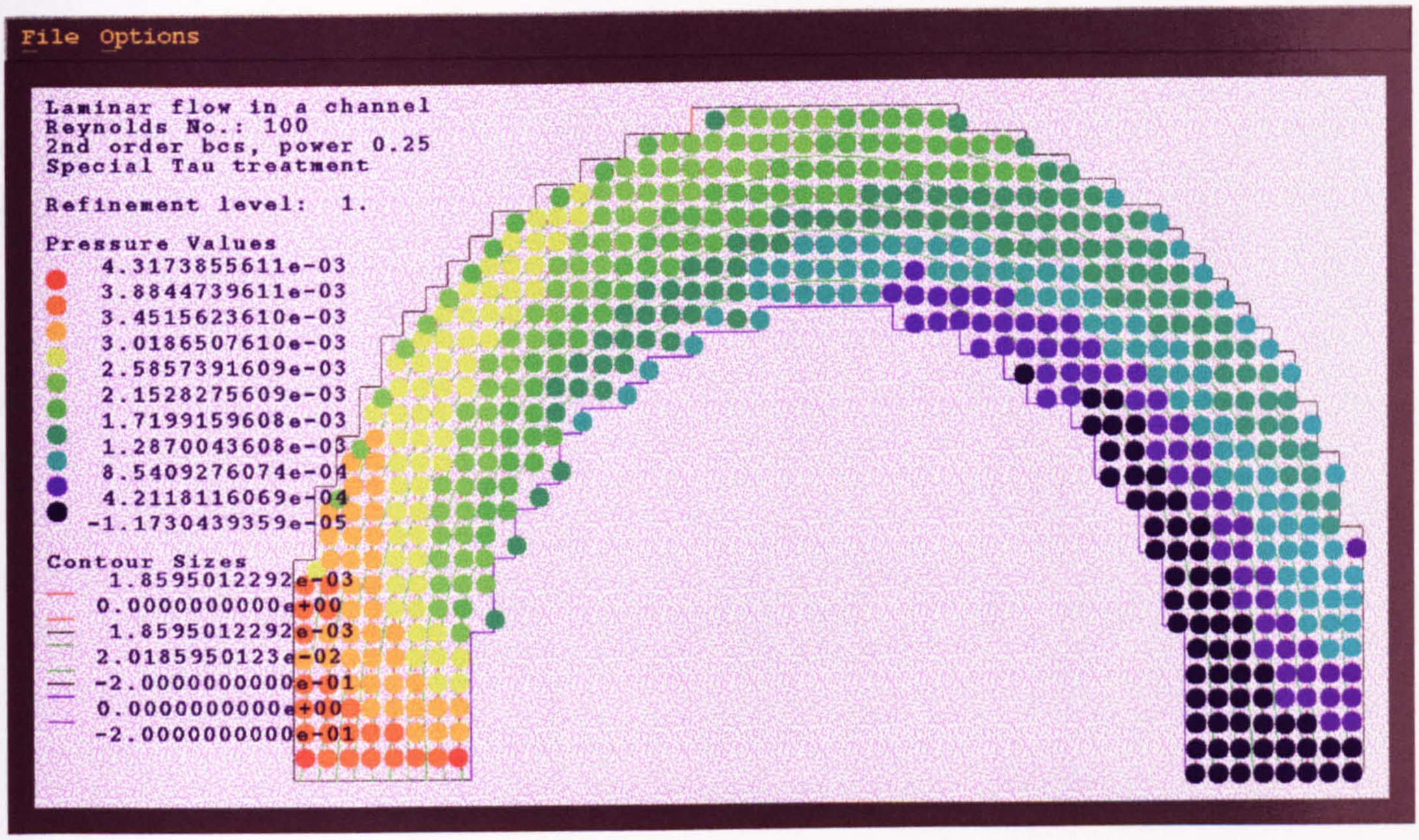


Figure 7.24: Semi-circle curved channel flow problem - Streamlines and pressure distributions for the CC-PAMG solution at Reynolds number = 100

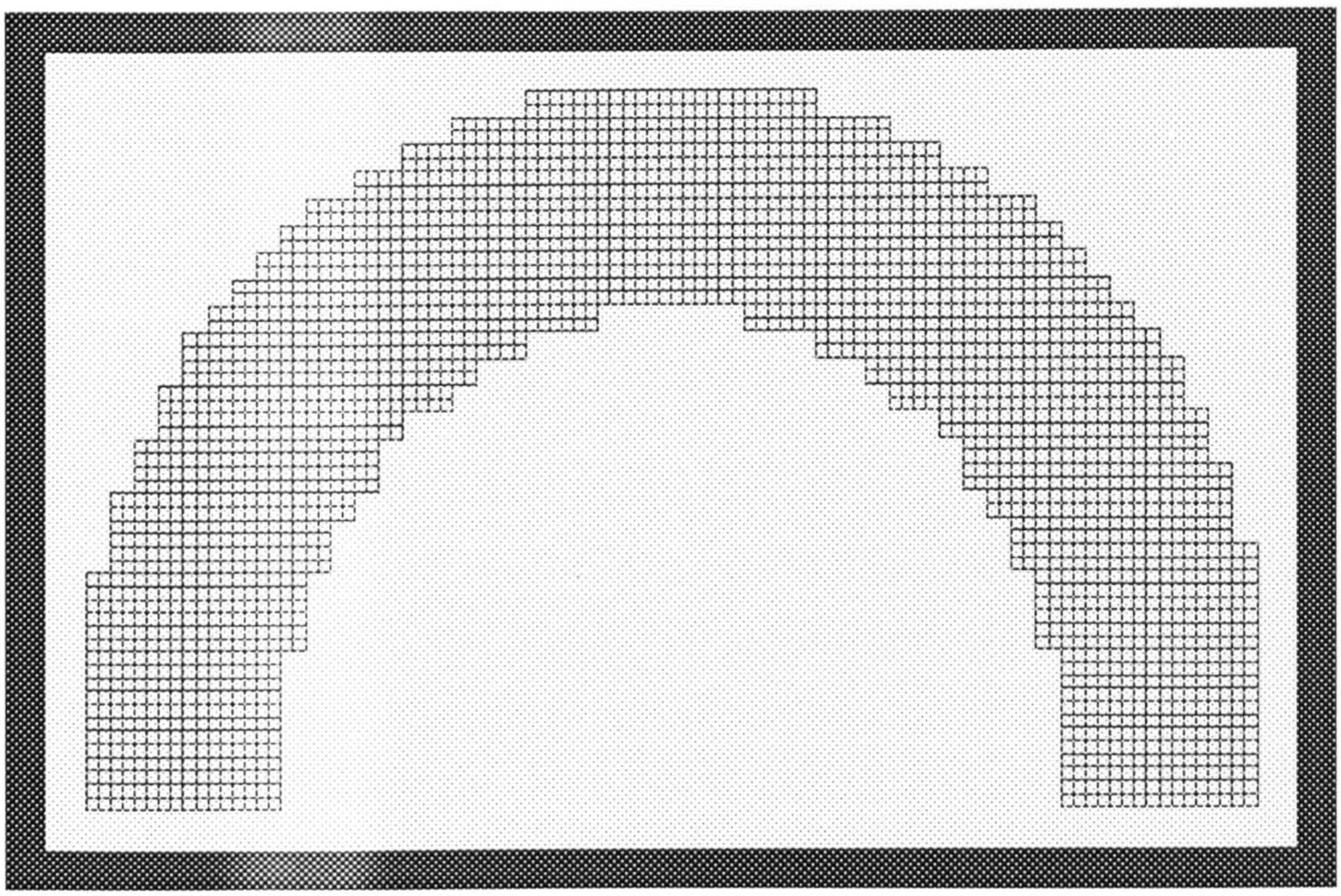


Figure 7.25: Semi-circle curved channel flow problem - Close-up of adapted grid at grid 1 for the CC-PAMG solution at Reynolds number = 100

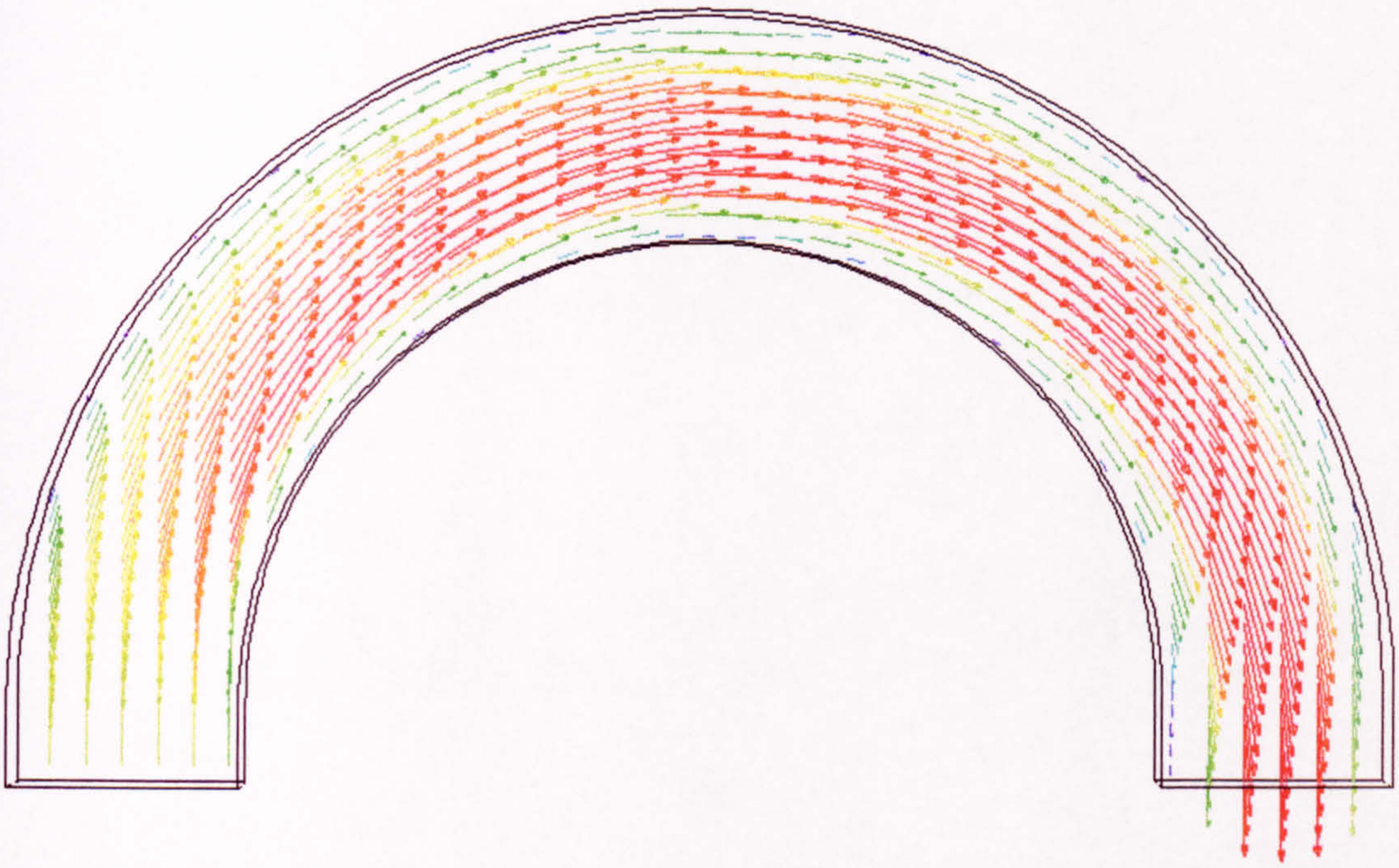


Figure 7.26: Semi-circle curved channel flow problem - Velocity vector distributions for the CFX 4.2 solutions at Reynolds number = 100

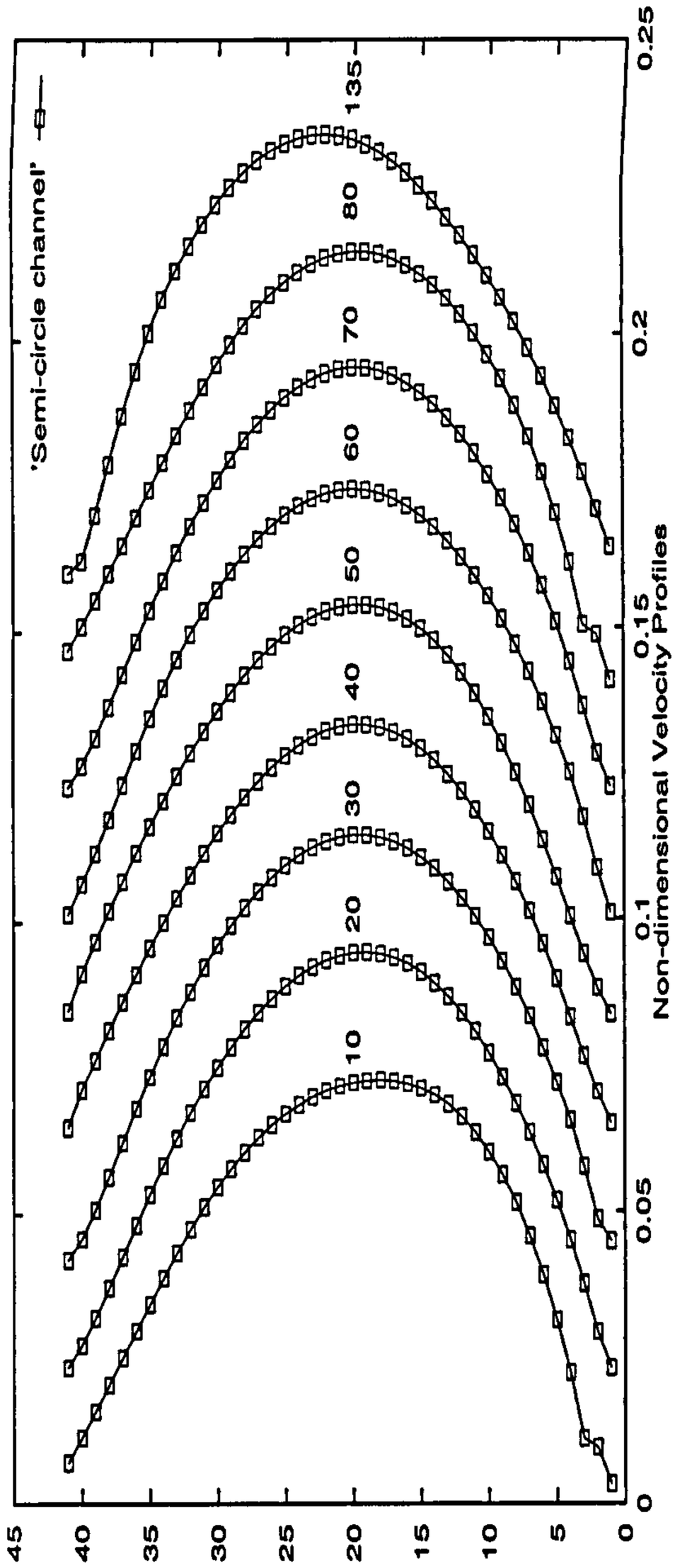


Figure 7.27: Semi-circle curved channel flow problem - Velocity profiles along the semi-circle curved zone for the CC-PAMG solutions at Reynolds number = 100

offset) direction; and then to the negative vertical direction (180^0 offset). It is clear that the velocity profiles near the bent walls are relatively small and smooth as shown in Figure 7.27. The velocity profiles are selected from 10^0 degree angle to 135^0 degree angle bended respectively.

Figures 7.28 to Figure 7.32 show the four levels of the velocity profiles calculated at five selected locations for each of the two solvers. Comparing the four levels of the velocity profiles at selected positions, it is shown that they match each other very well. The reason is that the grid size is small enough to deal with the curved boundaries. The results are very similar except in the boundary layers, the differences are observed near the walls. The refinement process has improved the accuracy in these areas.

As we expected, the CC-PAMG solutions and the CFX 4.2 results agreed well in the selected positions. Consider the fluid passing through a semi-circular curved channel, the pressure in the channel increases along the outside of the bend, which forces the flow to change direction gradually, and the maximum velocity in the core then shifts to the inside of the bend (see Figure 7.26 and Figure 7.29). The change in the pressure gradients at the bend appears clearly in Figure 7.24. The velocity vector distribution shows that the velocity increases in the curved channel and that the maximum velocity in the core shifts to the inside of the channel in most of the bend.

Comparing these two algorithms, it is evident that they are nearly the same in some selected positions. However, there are some significant differences: the maximum velocity in the core shifts slightly to the inside of the bend for the CFX 4.2 results at the 50^0 degree position, but the CC-PAMG solutions slightly shift to the outside of the channel in the same position; in the 0^0 degree curved position, the big difference occur in the middle of the channel. At this position, the flat inlet flow has not fully developed into a parabolic profile. Comparing the CC-PAMG solution and the results of CFX 4.2, our results have developed much better than the com-

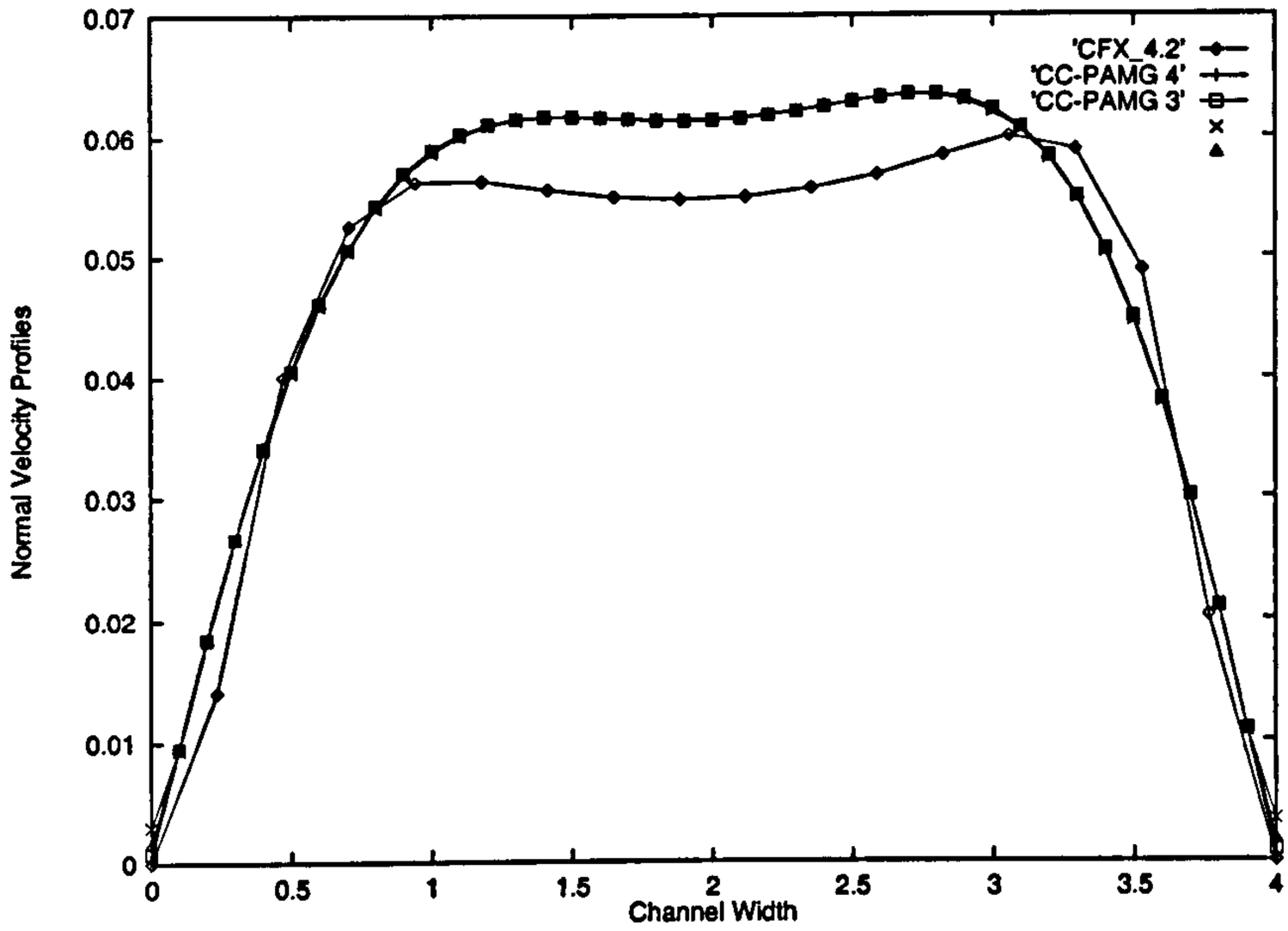


Figure 7.28: Semi-circle curved channel flow problem - Velocity profiles at zero degree bended. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

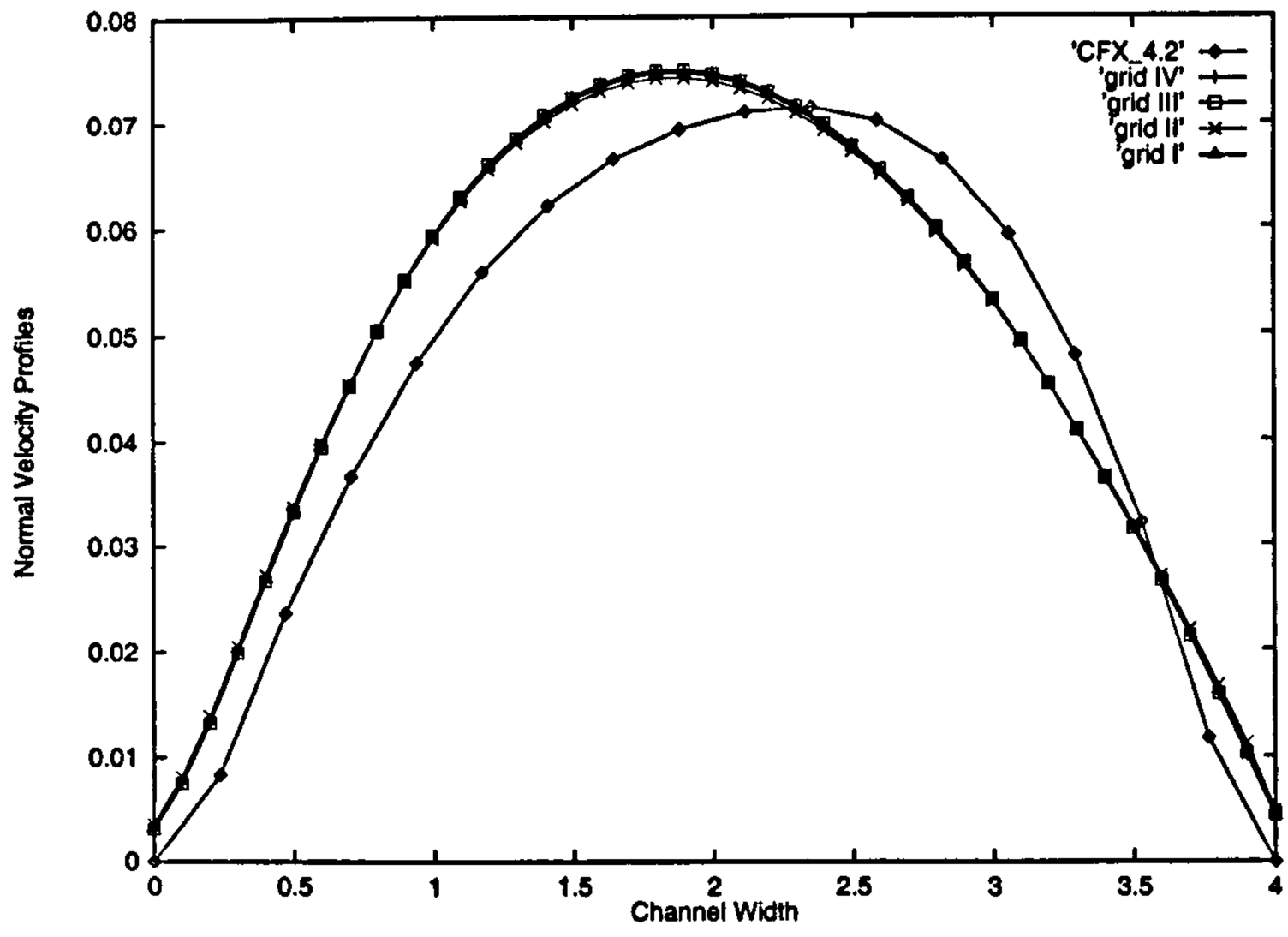


Figure 7.29: Semi-circle curved channel flow problem - Velocity profiles at position of 50 degree bend. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

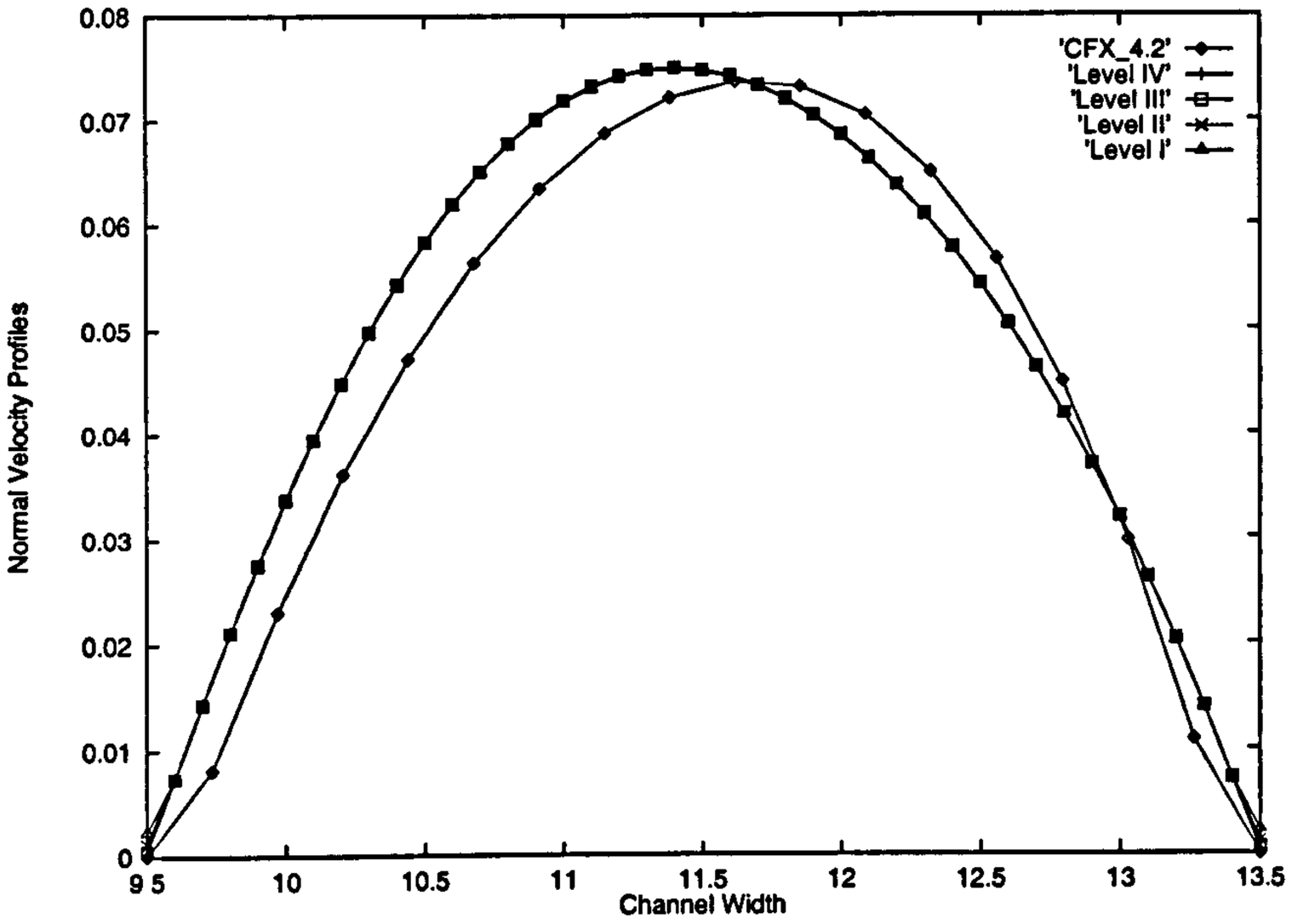


Figure 7.30: Semi-circle curved channel flow problem - Velocity profiles at 90 degree bend. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

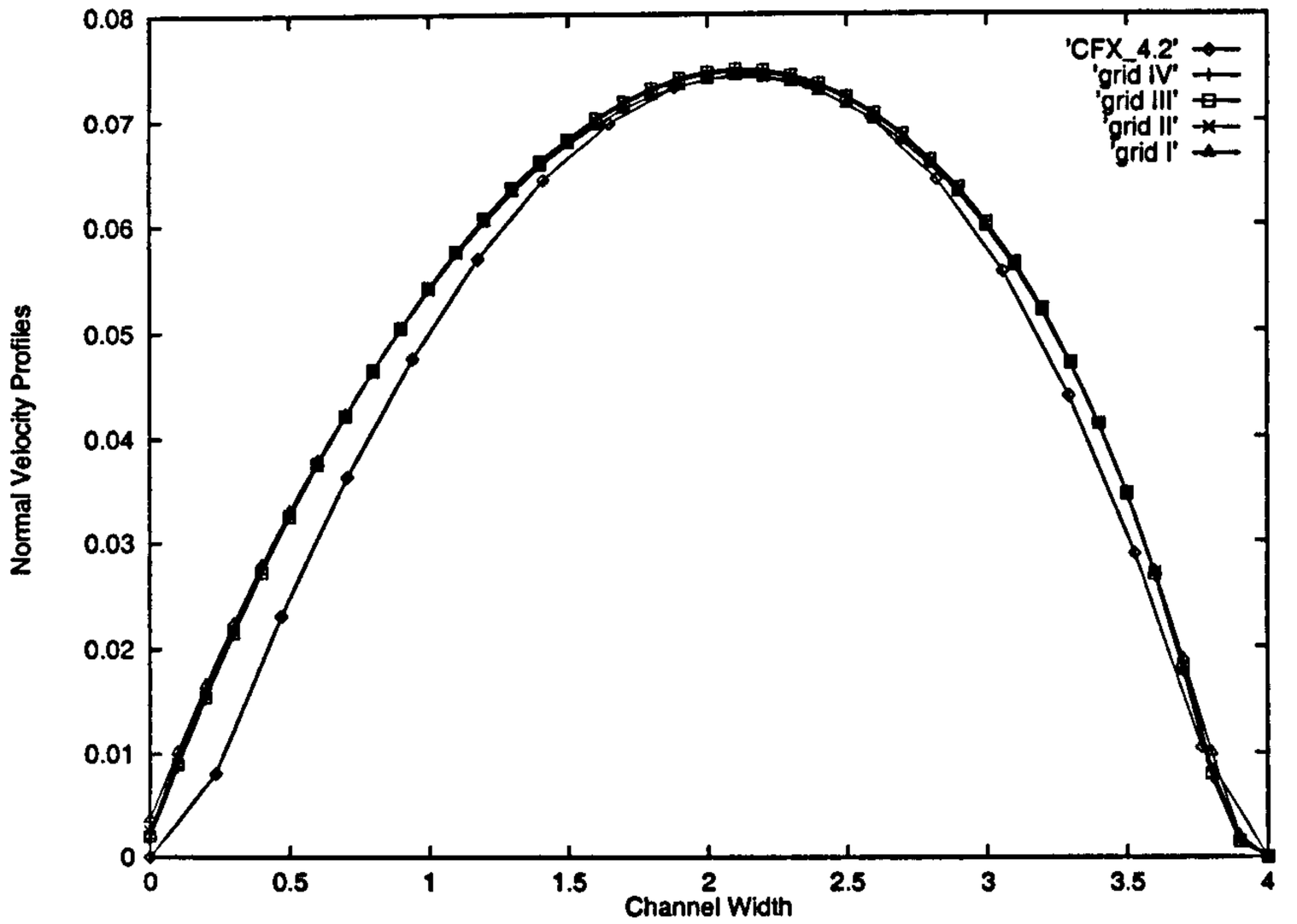


Figure 7.31: Semi-circle curved channel flow problem- Velocity profiles at 135 degree bended. Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

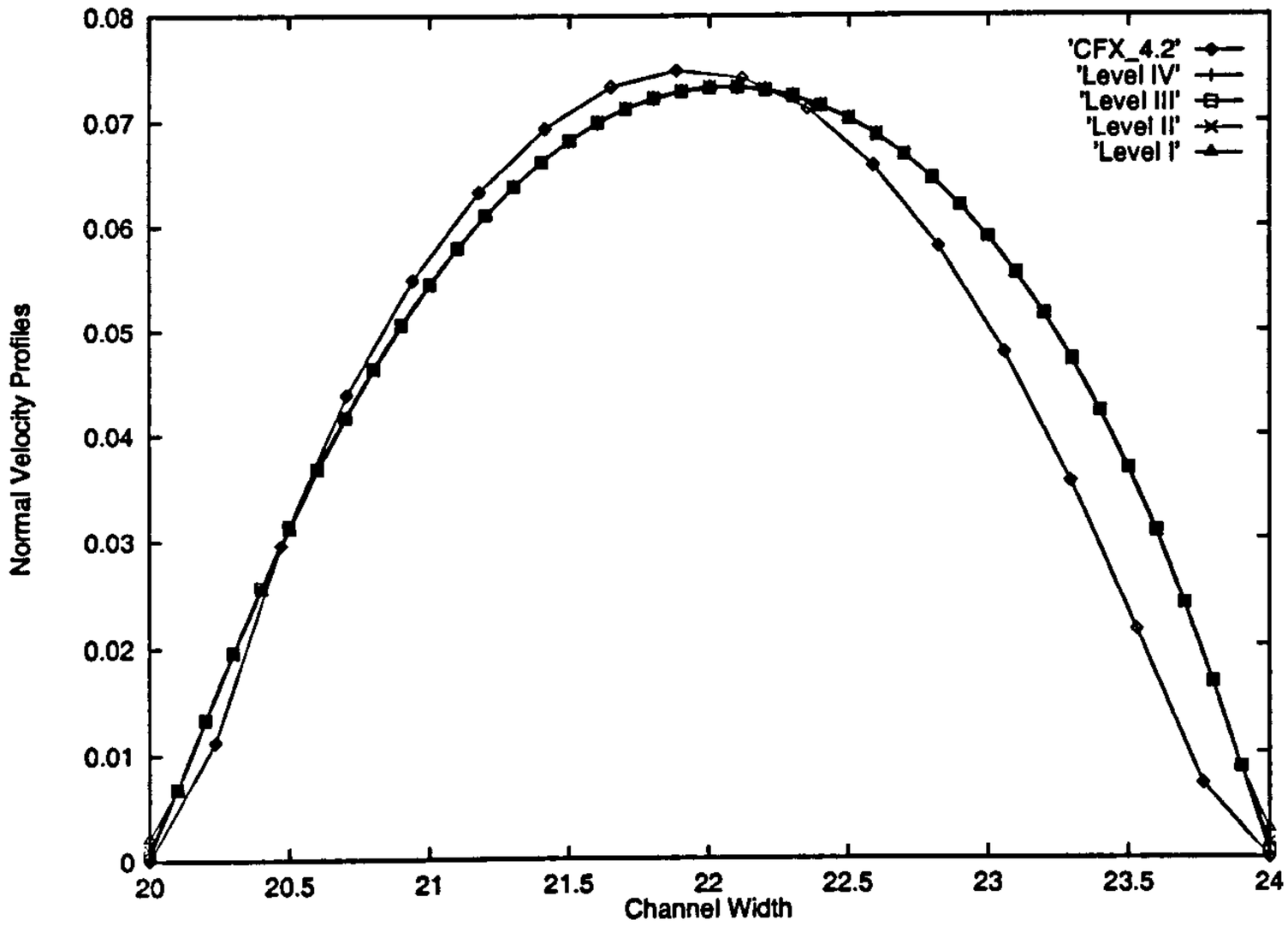


Figure 7.32: Semi-circle curved channel flow problem - Velocity profiles at 180 degree bend). Comparison of the CC-PAMG and CFX 4.2 solutions at Reynolds number = 100

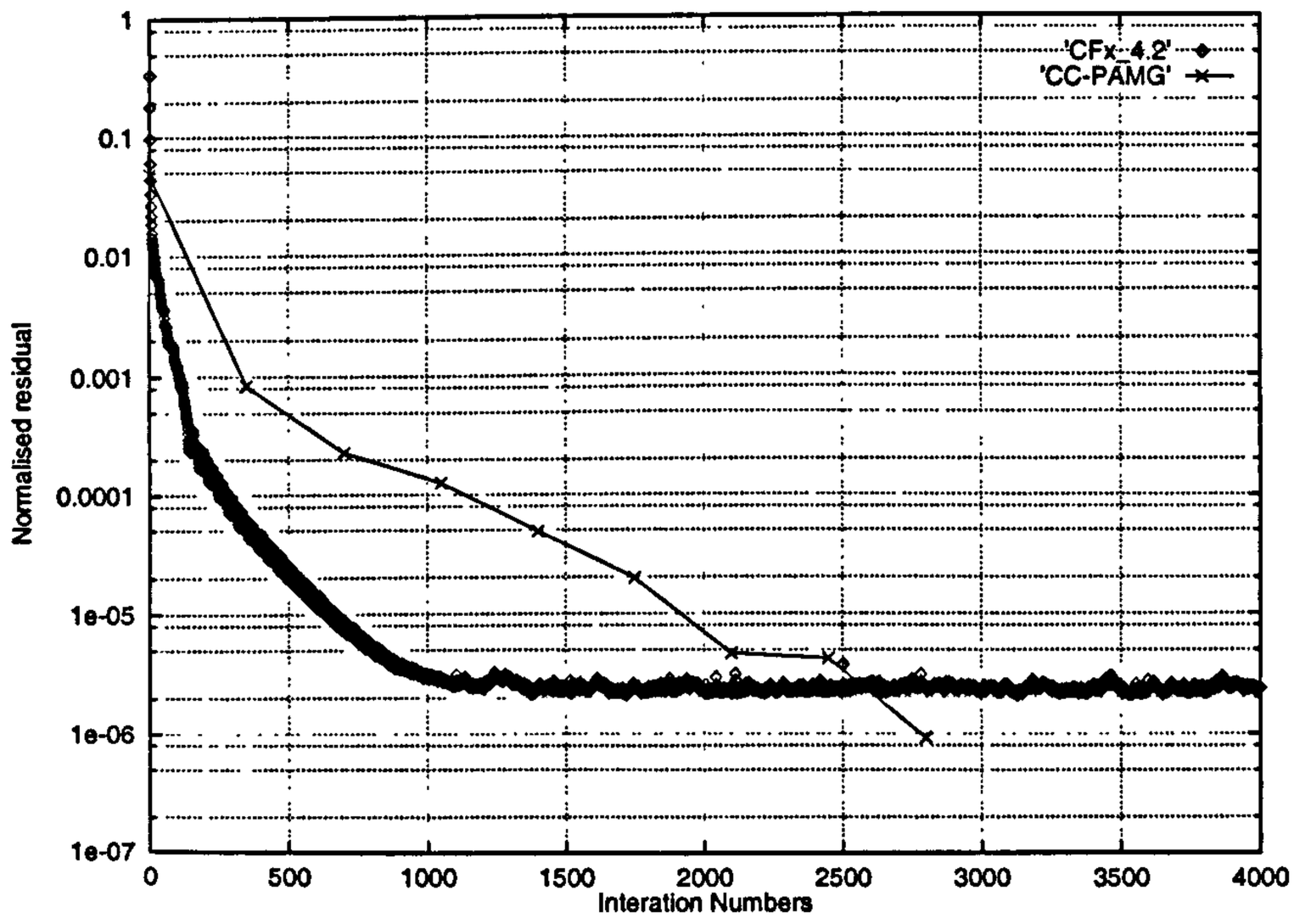


Figure 7.33: Semi-circle curved channel flow problem - Comparison of the convergence histories of CC-PAMG and CFX 4.2 at Reynolds number = 100

mercial package. Although the maximum velocity in the core shifts gradually along the channel, it always stands near the middle of the channel after coming into the bend. Notice the near wall profiles, the solution of the CC-PAMG are smooth and the solutions of CFX 4.2 change suddenly which may cause near-wall boundary errors. It can be improved by reduce the grid size as discussed in section 7.2.1

The Performance of the two Solvers

After investigation of the numerical solutions, the efficiency of the solvers and convergence factors have been compared with the Cartesian cut-cell algorithm. Numerical convergence is monitored by the reduction of the residual, measured using a suitable vector norm. The convergence factor has been used to investigate the performance of the solver.

In this section, we therefore concentrate on the convergence factors observed with CC-PAMG in the semi-circular channel flow. It should be noted that convergence factors are a large extent grid independent for the MG algorithm. This indicates that the multigrid procedures are performing well and that the slight relative degradation of convergence factors caused by specific features of the equations and boundary treatment. Comparison of the convergence histories are summarised by Figure 7.33. They clearly demonstrate that CC-PAMG out-performs a good segregated solver. It is shown that the CC-PAMG solver has a significant improvement of the convergence rate which saves computational times.

7.3 Conclusion

This Chapter documents results for the velocity profiles in curved channels using two different CFD algorithms. As is customary with this type of work, we have demonstrated the effectiveness and accuracy of our algorithm comparing our new results with a commercial algorithm.

A robust method has been developed whereby solid boundary conditions may be improved on a Cartesian mesh for arbitrarily shaped bodies. It has been combined with an adaptive mesh refinement scheme and a novel parallel multigrid method to produce a powerful algorithm for simulating incompressible flows in arbitrary geometries. Solutions are presented which demonstrate that this new algorithm can match not only the accuracy of the results produced using body-fitted grids (CFX 4.2 solutions), but also the geometric flexibility exhibited by unstructured grid schemes.

Chapter 8

Conclusion

In this thesis, we presented a new alternative approach for the prediction of steady incompressible flow fields involving complex two-dimensional geometries. The Cartesian cut-cell techniques and finite volume method have been used with the parallel adaptive multigrid algorithm. In this way, the CC-PAMG (cut-cell Parallel adaptive multigrid) approach deals with complex flows around complicated geometries. We bring this thesis to a close by presenting some conclusions that we have drawn from our work, together with a few suggestions as to how to our work could be usefully extended.

8.1 Summary

8.1.1 Algorithm

The main focus of the present work is to extend the original PAMG techniques to an unfitted grid formulation for internal and external flows. These concepts of PAMG have been implemented in a Navier-Stokes solver for steady viscous incompressible flows by Thompson et al.([118, 113, 117]). Whilst adaptive multigrid methods have been very successfully applied to body-fitted structure grid method and Cartesian grids for many years, this approach is entirely novel in that we apply it to incompressible flows with complex boundaries using Cartesian cut cell method. A new adaptive multigrid method called Cartesian cut-cell parallel adaptive multigrid method (CC-

PAMG) has been developed, building on the original solver PAMG. A range of original algorithmic developments have been introduced to solve these problems. They include:

- **Implementation of the Cartesian cut-cell method.**

In order to extend the PAMG solver to the unfitted grid formulation, the new Cartesian cut-cell method has developed to deal with the complex boundary conditions. A method for distinguishing cut-cell is introduced from analytical geometry. The investigation of error estimates for the cut-cells has been shown that the accuracy is second-order.

- **The derivation of consistent discretisation.**

Attention has to be devoted to interpolation operators which were required because of the use of staggered grids. Important features of the discretisation, the use of staggered finite volume grid and hybrid schemes are discussed for Cartesian grid and Cartesian cut cell grid and the discretisation process is described in detail. The second-order accuracy numerical schemes is achieved.

- **The modification of the adaptive multigrid transfers.**

The availability of the original PAMG solver was an advantage, but it's full exploitation has required a careful development for the Cartesian cut-cell with complex boundary conditions. The solver, prolongation, and restriction procedures have been carefully modified. The results presented so far, based on second-order accuracy prolongation of the pressure and third-order accuracy interpolation for the momentum equation corrections during the prolongation stages. The original PAMG has the first-order in pressure and the momentum equation corrections near the boundaries. Comparing the convergence of these two algorithms, the CC-PAMG is twice fast of the original PAMG solver (see Backward-facing step flows, Tables 5.5 and 5.6).

- **Accuracy investigation.**

Solutions obtained from CC-PAMG for a wide range of important test problems have been described, discussed in some detail and compared with analytical solutions (non-aligned channel flow and flow along a flat plate) and the solutions provided by independent software. We have chosen a widely used and validated commercial CFD package: CFX 4.2. CFX 4.2 solves the same set of governing equations as CC-PAMG but its solution algorithm is based on the SEMPLE procedure, which is segregated. A very good degree of agreement is obtained between the CC-PAMG and CFX 4.2 solutions for the bent channel. We have considered 90^0 and 180^0 bends. It can be considered that the implementation of the CC-PAMG solver for the incompressible flow with complex boundaries is essentially correct and accurate.

- **Robustness investigation.**

The robustness of the CC-PAMG solver was established by considering that its convergence is not dependent on the quality of the initial guess for the solution. For all the solutions presented in this thesis, uniform or parabolic initial distributions for all flow variables have been used. This is due to the fact that the use of hybrid schemes in a multigrid framework defines an implicit continuation method. By contrast, the performance of segregated solvers is very dependent on the quality of the initial guess.

8.1.2 Achievements

A computational method has been developed by which the solution of the steady-state two-dimensional Navier-Stokes Equations can be obtained using solution-based refinement for flows around bodies with arbitrary geometric complexity. This computational efficiency has been achieved by combining a high resolution finite volume scheme with an adaptive mesh strategy which includes automatic and spatial refinement.

A linked-list data structure is used to control the grids. Refinement on the basics

of flow and geometry allows better solution and geometry representation. The data structure is made up of a list of cells with pointers to a cell's parent, also 'aunts', neighbours and four children (three children if the parent cell is a cut cell, or one child or two if the parent cell is adaptively refined). Connectivity information is obtained from the data structure via this parent/child relationship.

The initial grid is generated with a minimum of user input for a complex configuration. The user need only specify a set of points defining the bodies, the base grid resolution, and cell size threshold for the initial grid. The grid is automatically adapted to the curvature of a body, providing the resolution required to resolve the body adequately.

Small cut cells are created by the arbitrary way that the Cartesian grid cuts through the body. This problem is easily overcome, by using a local cut cell treatment, coupled with a linear reconstruction method. This allows the solution to converge as robustly as if the small cells did not exist. The Navier-Stokes equations have been solved in conservation law using a finite volume formulation. The convective terms were treated using hybrid differencies. The solver was validated and its accuracy critically assessed by comparison to accepted computational results and to an exact solution of the Navier-Stokes equations.

In summary, the Cartesian cell based approach has been shown here to provide adaptively refined solutions to the Navier-Stokes equations with automatically generated grids. The mesh refinement was shown to improve the solution quality, also in an automated fashion. This demonstrated a new method, where adaptively refined solutions to the Navier-Stokes equations can be obtained upon complex domains with minimal user intervention. A series of staggered, viscous flux formulae have been investigated for use in the Cartesian cell based scheme. The assessments were made analytically, with specific focus on the accuracy and positivity of the schemes when used to construct discrete operators. The flux functions representative of linear and quadratic, were used to compute adaptively refined solutions of the Navier-Stokes

equations. The approach can yield excellent solutions and has been demonstrated to provide automatically generated solutions to the Navier-Stokes equations for complex domains at low and moderate Reynolds numbers.

8.2 Conclusion

One pressing need for CFD is the ability to generate, automatically and robustly, a grid for complex multi-dimensional configurations, and to compute stable and accurate flow solutions about that configuration. Many approaches are being used at the present, but no one method has emerged as the best option for grid generation and flow solution about arbitrarily complex geometries.

A Cartesian grid approach shows great promise as a method to overcome these problems. However Cartesian grids do have their share of difficulties to overcome. Any arbitrarily complex geometry can be cut from a Cartesian grid, but the resulting grid is a very poor representation of the body. A uniform Cartesian grid fine enough to resolve a body adequately requires many more cells than are needed away from the the body, thereby requiring unreasonable resources to obtain a solution. The solution to this problem is adaptation. A Cartesian grid which can be automatically and robustly adapted to a body to ensure that each of its features is adequately resolved is needed. Indeed, methods are being developed which are coming close to this ideal for fully arbitrary two-dimensional configurations. Especially promising is the reduction in total number of cells needed to create these grids. Cells can be added only where resolution of the body requires them. This is not only important in terms of physical memory usage to store the information, but also in terms of the real time required to generate the grid and compute flow solutions.

Automatic Cartesian grid generation is not the whole picture. The flow solution is also required. Here again, Cartesian grids have some problems. The somewhat arbitrary way that bodies are cut out of a Cartesian grid can create a huge disparity

in the sizes of neighbouring cells. This issue has been addressed to some degree in two-dimensions in this thesis, but questions remain about how well these methods will carry over to 3-dimensions. Flow solution methods for unstructured grids is currently a widely researched topic, and new innovations may have a large effect on the effectiveness of flow solution on Cartesian grids. One huge benefit to the flow solution using a Cartesian grid is that grid adaptation has already been developed in creating the grid. Using additional grid adaptation by solution-based information will provide more accurate solutions for less cost.

The results of the method presented in this thesis show that this Cartesian grid approach is viable. Second-order global accuracy is obtained for the given computed solutions. These solutions compare very favourable not only with the CFX 4.2 benchmark solutions, but also with exact incompressible solutions. The number of cells required to obtain these solutions is greatly reduced due to adaptation. In fact, comparable structured grid solutions require two to ten times the number of cells needed for the Cartesian method. Overall, the flow solutions computed with this Cartesian approach with less cells to compute, with the same solution accuracy and reliability.

Test case	Reynolds No.	Comparison
Flat Plate	100, 10,000	Analytic/Ru-Ku/PAMG
Back-step	100, 500	PAMG
Non-aligned Channel	$1 \sim 500$	Analytic
90^0-bend Channel	50, 100, 500	CFX 4.2
180^0-bend Channel	100, 500	CFX 4.2

Table 8.1: List of the test cases comparison with analytical solution and other numerical results.

At the close of this project, it can reasonably claim that the initial goals of designing a robust, efficient and accurate algorithm for the simulation of incompressible flow

with relatively complex geometry have been achieved. Adaptive multigrid provides very significant acceleration and finally adaption greatly reduces the computational cost of obtaining a solution to a given accuracy.

8.3 Future Work

There are number of areas of research where this work could be naturally extended. The first extension is to unsteady flow. The most obvious approach is to merge cut cells with complex boundary conditions. Any cell which is cut by the body would be merged with one of its un-cut neighbour cells. This results in a slight accuracy loss on the body, but the time-step from the neighbour cell is used rather than the small time-step from the cut cell. This does not address the disparate time-steps from cells on different grid levels, where more sophisticated time-step schemes are needed [26].

Another likely application of this work is to simulate multiphase flows. The main difficulty here is that the governing equations are mathematically complex and their solutions are subject to constraints. The solver has therefore required a considerable number of specific algorithmic developments for the complex geometry. The discretsation of the governing equations is also not a trivial task, and the simulation of two-phase flows has been developed in successfully in rectangular boundaries for adaptive multigrid method (see P. Lezeau [63]). In fact, the work in this thesis could be extended to any set of conservation equations, including internal and external flows.

The most obvious extension to this work comes in extending the algorithm to three dimensions. The appropriate data structure in the three-dimensional case is an oct-tree, with each parent cell being divided into eight child cells. The body definition is one obvious difficulty to overcome. Defining a body surface from a set of points and incorporating that information into a code is a difficult task. Memory usage and computer speed become much more important as well.

Appendix A

Multigrid Method

A.1 Basic Multigrid Principles

Multigrid methods can simply be seen as an acceleration technique for the iterative solution of algebraic systems of equations, or in their full generality, as a consistent framework enabling the solution of a given problem to be found at a nearly optimal cost. The basic idea is to define a sequence of grid, and to cycle through the different grids in the course of the computations so that the features of the solution are resolved on the most suitable grid.

The basic principles involved in the construction of a multigrid algorithm are discussed in this section. These principles are basic in that they do not depend on the particular set of equations being solved, the discetization and types of grids employed, or the dimensionality of the problem. These principles are then utilized to demonstrate the construction of a exemplary multigrid algorithm for solving the flow equations.

A.1.1 Multigrid Correction Scheme

The simplest way to describe a multigrid scheme is to start with a two-level scheme for linear problems. Consider the solution of the discrete problem:

$$L^h u^h = f^h \tag{A.1}$$

where we assume that L is a linear elliptic operator; u is the unknown; f is the source term and the superscript ($.^h$) is used to refer to operators and grid functions defined on a mesh with spacing h. The current estimate of the solution u^h is denoted as $\bar{u}^h$, which is obtained by approximately solving the above equation, using an iterative technique. Since $\bar{u}^h$ does not satisfy the above equation exactly, its substitution into equation (1) yields

$$L^h \bar{u}^h - f^h = r^h \tag{A.2}$$

where r^h is called the residual, and vanishes only when the exact solution to the discrete problem is attained. The object of the multigrid scheme is to compute a correction v^h such that the exact solution is given by:

$$u^h = \bar{u}^h + v^h \tag{A.3}$$

Taking the difference of equation (A.1) and (A.2), one obtain

$$L^h u^h - L^h \bar{u}^h = -r^h \tag{A.4}$$

If the operator L^h is linear, the above equation may be reduced to an equation for the correction v^h by substituting equation (A.3) into (A.4), which yields

$$L^h v^h = -r^h \tag{A.5}$$

Assuming that the high-frequency errors in the solution have been eliminated by sufficient fine grid smooth cycles, the remaining correction v^h which we seek must be smooth, and can therefore be computed more efficiently on a coarser grid by solving the equation

$$L^H v^H = -I_h^H r^h \tag{A.6}$$

Where the superscript H denotes a coarser grid, and I_h^H is an operator which interpolates residuals from the fine grid h to the coarse grid H. I_h^H is usually called the

restriction operator.In the event, the present procedure may be performed recursively on coarser grids, thus yielding an approximate solution to the above equation. The exact or approximate solution of v^H must then be employed to correct the original fine grid solution. This is achieved as

$$\bar{u}^h_{new} = \bar{u}^h + I^h_H v^H \tag{A.7}$$

where I^h_H, which represents the interpolation of the coarse grid corrections v^H to the fine grid, is called the **prolongation** operator. Once these fine grid values have been updated, they may be smoothed again by additional fine grid iterations, and the entire procedure, which constitutes a single multigrid cycle, may be repeated until overall convergence is obtained. This particular multigrid strategy is called the **Correction Scheme**, since the coarse grid equations operate directly on the correction variables v^h.

A.1.2 Full Approximation Storage Scheme

Full approximation storage (FAS) is a natural modification of the Correction Scheme for nonlinear problems. We introduce an approximation to the solution on the coarse grid variable $\bar{u}^H$ defined as

$$\bar{u}^H = \bar{I}^H_h \bar{u}^h + v^H \tag{A.8}$$

where $\bar{I}^H_h$ represents an operator which interpolates fine grid solution variables to the coarse grid. The coarse grid equation equivalent to equation (A.6) can now be written as

$$L^H \bar{u}^H - L^H \bar{I}^H_h \bar{u}^h = -I^H_h r^h \tag{A.9}$$

As previously, I^H_h represents the restriction operator which transfers residual from fine to coarse grids. The operator $\bar{I}^H_h$ and I^H_h may in principle be different from one another.

It is useful to rewrite the above equation as

$$L^H \bar{u}^H = S^H \tag{A.10}$$

where,

$$S^H = L^H \bar{I}_h^H \bar{u}^h - I_h^H r^h \tag{A.11}$$

In the above form, the coarse grid equation is seen to take on a similar structure to the original fine grid equation, which a modified source term. This enables the use of similar techniques for solving the coarse and fine grid problems. Once the coarse grid equations have been solved, either exactly or approximately, the fine grid variables are updated as follows:

$$\bar{u}_n ew^h = \bar{u}_o ld^h + I_H^h(\bar{u}^H - \bar{I}_h^H u^h) \tag{A.12}$$

The source term S^H may also be rewritten as:

$$S^H = f^H + \tau^H \tag{A.13}$$

where

$$f^H = I_h^H f^h \tag{A.14}$$

and

$$\tau^H = L^H \bar{I}_h^H \bar{u}^h - I_h^H(L^h u^h) \tag{A.15}$$

τ^H is sometimes called the defect correction. It can be loosely described as the difference between the coarse grid discritization and the interpolation of the fine grid discritization onto the coarse grid.

The ability to directly handle non-linear problems is one of the great advantages of multigrid algorithms. This obviates the need to linearize the problem, with the possible complications and overheads which such operations often entail.

A.2 Multigrid Cycling Strategies

Cycling strategies refer to techniques employed to determine when to switch from one grid to the next. These can be divided into basic approaches: adaptive and fixed cycling strategies. Adaptive cycling methods involve the monitoring of the numerical

convergence process. When it is determined that the high-frequency errors on the current grid have been effectively eliminated, usually by observing a sharp slowdown in the convergence rate, the jump to a coarser grid is triggered. Although adaptive cycling strategies may appear more desirable, practical considerations such as simplicity and robustness usually result in the use of fixed cycling strategies, where a fixed pattern of coarse and fine grid iterations is prescribed. The four most common cycling patterns are the $V-cycle, W-cycle, F-cycle$, and Full MultiGrid Method (FMG).

The multigrid $V-cycle$ begins on the finest grid of the sequence, where one relaxation or time-step is performed. The solution and residual are then interpolated to the next coarser grid, where another time-step is performed. This procedure is repeated on each coarser grid until the coarsest grid of the sequence is reached. At this point, the coarse grid corrections are prolongated back to each successively finer grid until the finest grid is reached.

An alternative to the $V-cycle$ is to apply the two-level algorithm twice in order to solve the coarse grid problem. This results in the $W-cycle$ (see Figure A.2). The $W-cycle$ is a recursive strategy which weights coarse grids more heavily. Use of the $W-cycle$ is often found to be more efficient overall, and more robust the $V-cycles$ for non-linear problems.

The third cycling strategy is the $F-cycle$ (see Figure A.3). It is quit similar to the $W-cycle$ in the sense that coarse grid computations are repeated in order to improve the quality of the coarse grid correction. It is less costly than the $W-cycle$ strategy.

Another approach to accelerating the convergence of iterative methods is to provide good starting guesses. The combination of mesh sequencing with a multigrid method results in a strategy known as the Full MultiGrid (FMG) method. The basic cycling strategy is depicted in Figure A.4, using a saw-tooth multigrid $V-cycle$. be-

ginning with an initial sequence of grids, the solution on the finest grid of the sequence is obtained by a multigrid procedure operating on this sequence. A new finer grid is then added to the sequence, the solution is interpolated onto this new mesh, and multigrid cycling is resumed on this new larger sequence of meshes, The procedure can be repeated, each time adding a new finer grid to the sequence, until the desired solution accuracy has been achieved, or the finest available mesh has been reached.

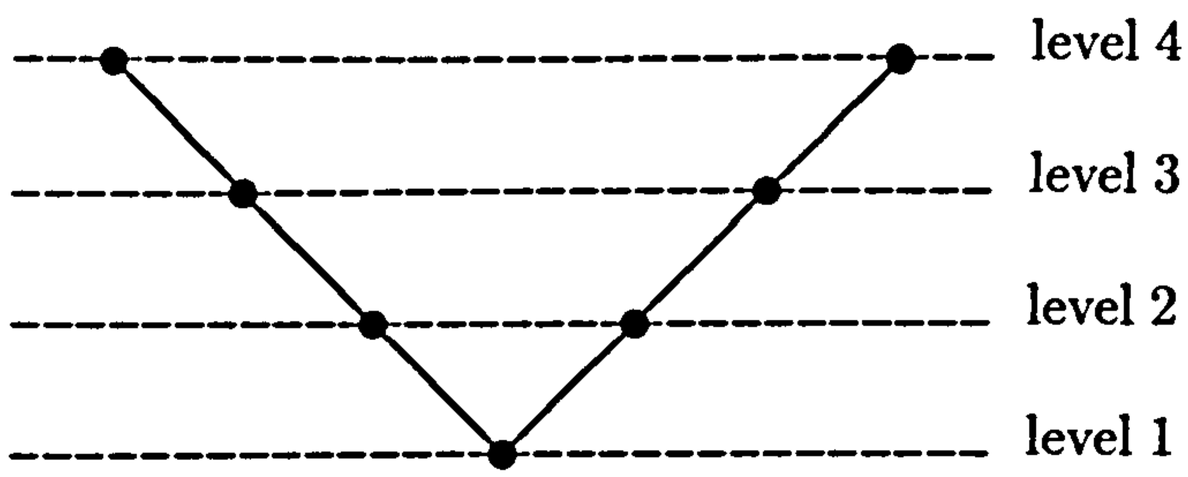


Figure A.1: Four level multigrid V-Cycle

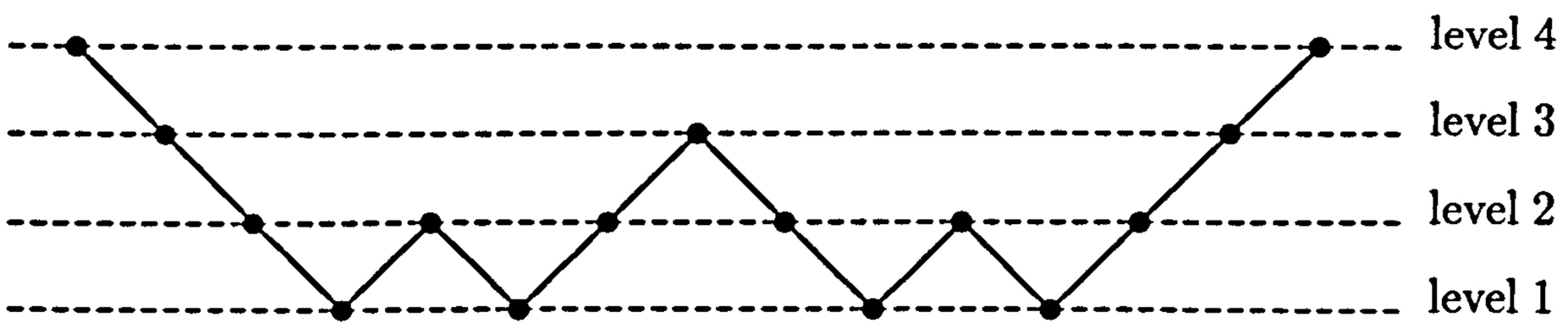


Figure A.2: Four level multigrid W-Cycle

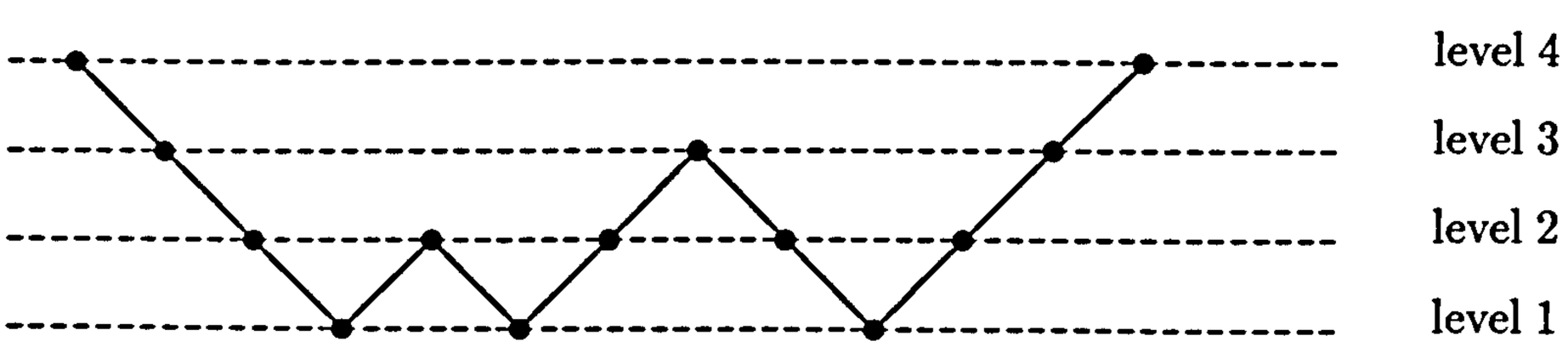


Figure A.3: Four level multigrid W-Cycle

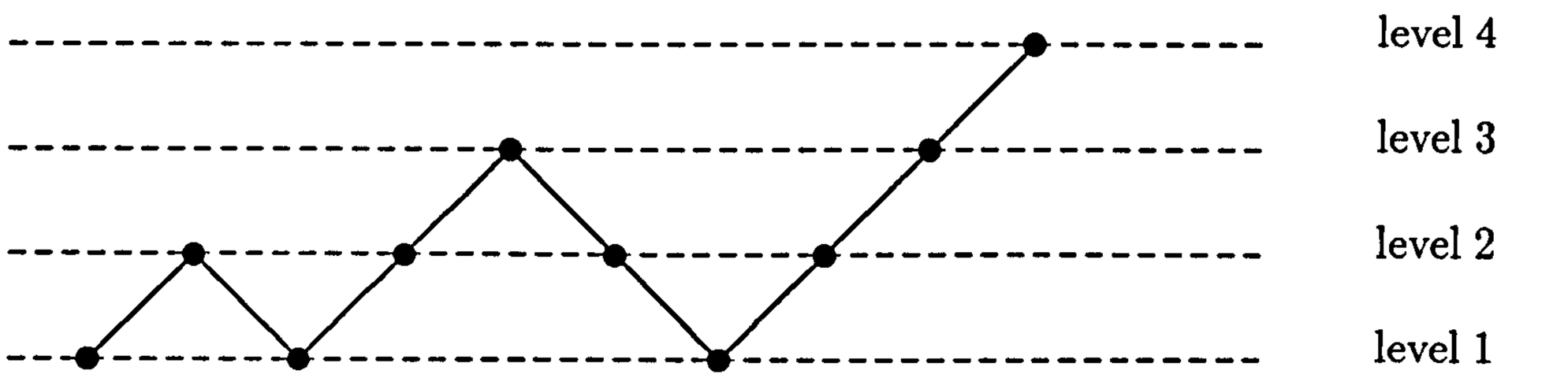


Figure A.4: Four level multigrid FMG algorithm

A.3 Multigrid Algorithm for the Navier-Stokes Equations

The remaining procedure which must be specified in the construction of a multigrid algorithm is the single relaxation . The requirements of this procedure are that it be capable of rapidly damping out high-frequency errors. While the convergence of almost any relaxation scheme may be accelerated through a multigrid procedure, the most effective multigrid strategy is to rely on a simple and inexpensive relaxation scheme, which is specifically tailored for annihilating high-frequency errors. All lower frequency errors can then be handled effectively by the appropriate coarse-grid level. The use of multi-stage scheme as multigrid drives is one of the most common approaches for both structured and unstructured mesh problems.

Having described all essential elements of the multigrid strategy, a particular multigrid algorithm for the Navier-Stokes equations can now be described. The algorithmic steps are as follows:

Full Multigrid (FMG) Method

$$V^h \leftarrow FMG^h(V^h, f^h) \tag{A.16}$$

1. If Ω^h = coarsest grid, then go to step 3.

Else

$$f^{2h} \leftarrow I_h^{2h}(f^h - A^h V^h) \tag{A.17}$$

$$v^{2h} \leftarrow 0 \tag{A.18}$$

$$v^{2h} \leftarrow FMG^{2h}(v^{2h}, f^{2h}). \tag{A.19}$$

2. correct $V^h \leftarrow V^h + I_{2h}^h V^{2h}$.
3. $V^h \leftarrow MV^h(V^h, f^h)$ ν_0 times.

Where $MV^h(V^h, F^h)$ is V-Cycle Scheme. More details of the multigrid algorithm can be found in the bibliography.

Appendix B

Accuracy of Non-uniform Grids

B.1 Non-uniform Grids

For the sake of simplicity, most examples have focused on uniform grids of nodal points in the past. However, the derivation of discretised equations in Chapter 3 and 4 used general geometrical dimensions and is also valid for non-uniform grids. In a non-uniform grid the faces e and w of a general node may not be at the mid-points between nodes E and P, and nodes W and P, respectively. In this case the interface values of diffusion coefficients Γ are calculated as follows:

$$\Gamma_w = (1 - f_W)\Gamma_W + f_W\Gamma_p \tag{B.1}$$

where the interpolation factor f_W is given by

$$f_W = \frac{\Delta_{Ww}}{\Delta_{Ww} + \Delta_{wP}} \tag{B.2}$$

and

$$\Gamma_e = (1 - f_P)\Gamma_P + f_P\Gamma_E \tag{B.3}$$

where

$$f_P = \frac{\Delta_{Pe}}{\Delta_{Pe} + \Delta_{eE}} \tag{B.4}$$

For a uniform grid these expressions simplify to equation, since $f_W = 0.5$, $F_P = 0.5$, we have $\Gamma_w = (\Gamma_W + \Gamma_P)/2$ and $\Gamma_e = (\Gamma_P + \Gamma_E)/2$. According to the discussion

in section 4.5.1, we know that has a second-order approximation for the non-uniform stepsize expressions.

Basically there are two practices used to locate control volume faces in non-uniform grid interpolations.

B.1.1 First Practice

Nodal points are defined first and the control volume faces located mid-way between the grid points. This is illustrated in Figure B.1.

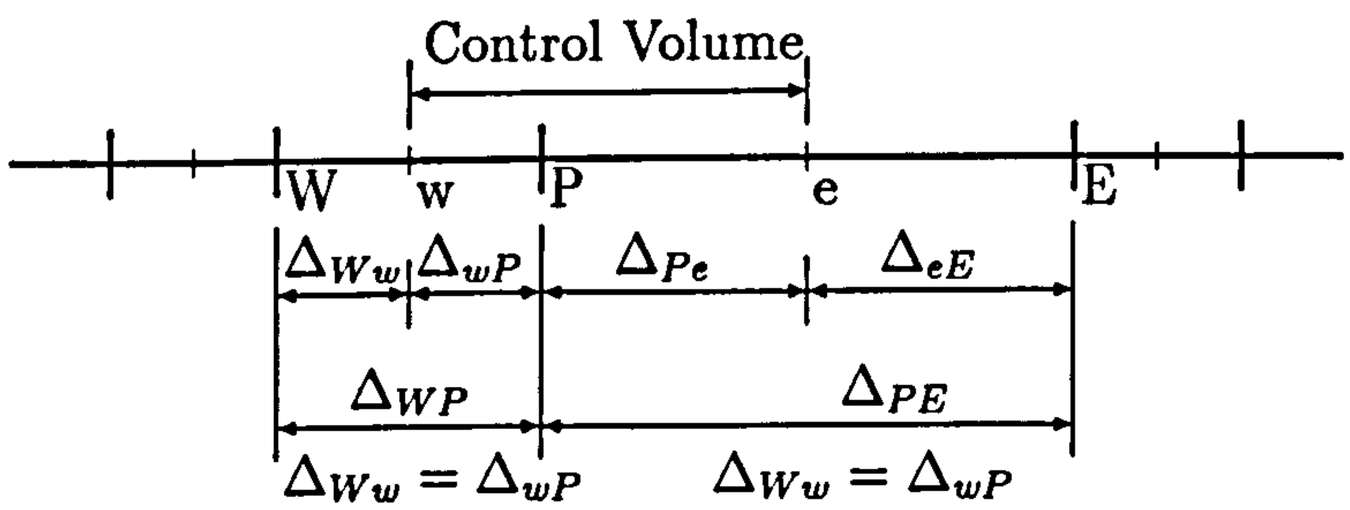


Figure B.1: Practice 1 Non-uniform grids illustration.

B.1.2 Second Practice

Locations of the control volume faces are defined first and the nodal points are placed st the centres of the control volumes. This is illustrated in Figure B.2. Here the faces of a control volume are not at the mid-point between the nodes. The evaluation of gradients obtained through a linear approximation is unaffected because the gradient remains the same at any point between the nodes in question but the values of diffusion coefficient Γ need to be evaluated using interpolation function (1).

It is very important to note that central difference formulae for the calculation of gradients at cell faces and the scheme for convective fluxes are only second order

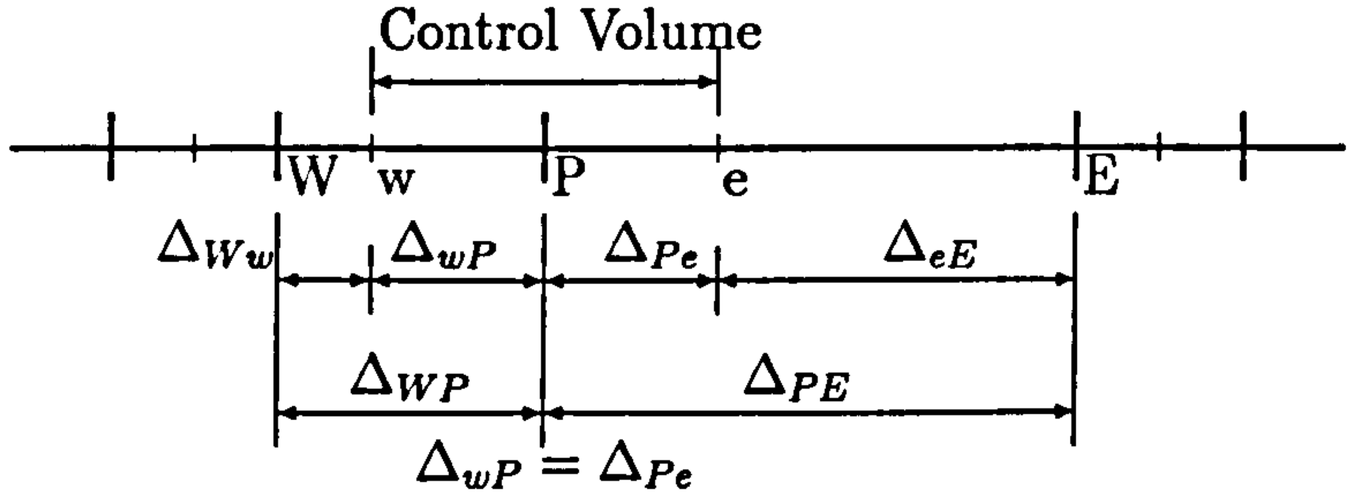


Figure B.2: Practice 2 Non-uniform grids illustration.

accurate when the control volume face is mid-way between the nodes. In practice a control volume face, e for example, lies mid-way between nodes P and E, so the differencing formula used to evaluate the gradient $(\partial\phi/\partial x)_e$ is second -order accurate. A further advantage of evaluate of first practice is that property values Γ_e, Γ_w etc. can be easily evaluated by taking the averaged values. The disadvantage of first practice is that the value of the variable ϕ at P may not necessarily be the most representative vale for the entire control volume as point P is not at the centre of the control volume. In practice the value of ϕ at P is a good representative value for the control volume as P lies at the centre of the control volume, but the discretisation schemes lose accuracy. A thorough discussion on these two practices is found in Jones and Thompson [58] for more details.

Bibliography

[1] N. S. Bachvalov, *On the convergence of a relaxation method with natural constraints on the elliptic operator*, USSR Comput. Math. and Math. Phys. **6** (1966), 101–135, translation.

[2] D. Bai and A. Brandt, *Local mesh refinement multilevel techniques*, SIAM J. Sci. Stat. Comput. **8** (1987), no. 2, 109–134.

[3] T. J. Barth, *On unstructured grids and solvers*, Computational Fluid Dynamics, vol. Lecture Series 1990-04, Von Karman Institute for Fluid Dynamics, 1990.

[4] ———, *Recent developments in high order k-Exact reconstruction on unstructured mesges*, 31st Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 11–14 1993, AIAA 93-0668, pp. 1–15.

[5] T. J. Barth and P. O. Frederickson, *Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction*, AIAA 90–0013, AIAA, January 1990.

[6] T. J. Barth and D. C. Jespersen, *The design and application of upwind schemes on unstructured meshes*, AIAA 89–0366, AIAA, January 1989.

[7] J. Bell, M. Berger, J. Saltzman, and M. Welcome, *Three-dimensional adaptive mesh refinement for hyperbolic conservation laws*, SIAM J. Sci. Comput. **15** (1984), no. 1, 127–138.

[8] J. B. Bell, P. Colella, and J. A. Trangenstein, *Higher order godunov methods for general systems of hyperbolic conservation laws*, Journal of Computational Physics **82** (1989), 362–397.

[9] M. J. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, Journal of Computational Physics **82** (1989), no. 1, 64–84.

[10] M. J. Berger and A. Jameson, *Automatic adaptive grid refinement for the Euler equations*, AIAA Journal **23** (1985), no. 4, 561–568.

[11] M. J. Berger and R. J. LeVeque, *Stable boundary conditions for Cartesian grid calculations*, ICASE Report No. 90-37/NASA Contractor Report 182048, NASA, Langley Research Center,Hampton, Virginia, May 1990.

[12] ———, *A rotated difference scheme for Cartesian grids in complex geometries*, Aiaa paper 91 – 1602, AIAA, 1991.

[13] M. J. Berger and J. Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, Journal of Computational Physics **53** (1984), 484–512.

[14] A. Brandt, *Multi-level adaptive techique (mlat) for fast numerical solution to boundary value problems*, Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics (Paris,1972) (H. Cabannes and R. Temam, eds.), vol. 1, Springer-Verlag, Berlin, 1973, Lecture Notes in Physics, Vol. 18, pp. 82–89.

[15] ———, *Multi-level adaptive solutions to boundary-value problems*, Mathematics of Compuation **31** (1977), no. 138, 333–390.

[16] ———, *Guide to multigrid development*, Multigrid Methods (W. Hackbusch and U. Trottenberg, eds.), Lecture Notes in Mathematics, vol. 960, Springer, Berlin, 1982, Multigrid Methods Proceedings, pp. 220–312.

[17] ———, *Multigrid techniques: 1984 guide –with applications to fluid dynamics*, Computational Fluid Dynamics, 1984–04, von Karman Institute, Belgium, 1984.

[18] ———, *Multilevel computations: Review and resent developments*, Multigrid Methods (S. F. McMormick, ed.), Lecture Notes in Pure and Applied Mathematics, vol. 110, Marcel Dekker, New York, 1988, pp. 35 – 62.

[19] ———, *The Weizman institute research in multilevel computation:1989 report*, Proc. 4th Copper Mountain Conference on Multigrid Methods (Philadelphia) (J. Mandel, S. F. McMormick, Jr. J. E. Derdy, C. Farhat, G. Lonsdale, S. V. Parter, J. W. Ruge, and K. Stüben, eds.), SIAM, 1989, pp. 13 – 53.

[20] A. Brandt and N. Dinar, *Multigrid solutions to elliptic flow problems*, Numerical Methods For Partial Differential Equations (Madison,Wisconsin) (S. V. Parter, ed.), Mathematics Research Center, The University of Wisconsin, Academic Press, October 23–25 1978, pp. 53–147.

[21] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, Journal of Computational Physics **90** (1990), 348 – 370.

[22] W. L. Briggs, *A multigrid tutorial*, Society for Industrial and Applied Mathematics, 1987.

[23] J. C. Chai, G. Parthasarathy, S. V. Patankar, and H. S. Lee, *A finite volume radiation heat transfer procedure for irregular geometries*, 6th AIAA/ASME Joint Thermophysics and Heat Transfer conference (Colorado Springs, Co), JUNE 20–23 1994, AIAA 94 – 2095, pp. 1–8.

[24] G. Chessire and W. D. Henshaw, *Composite overlapping meshes for the solution of partial differential equations*, Journal of Computational Physics **90** (1990), 1–64.

[25] Y. Chiang, B. van Leer, and K. G. Powell, *Simulation of unsteady inviscid flow on an adaptively refined Cartesian grid*, 30th Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 6 – 9 1992, AIAA-92-0443, pp. 1–16.

[26] Y. L. Chiang, *Simulation of unsteady inviscid flow on an adaptive refined cartesian grids*, Ph.d. dissertation, The university of Michigan, Michigan,USA, 1992.

[27] D. K. Clarke, M. D. Salas, and H. A. Hassan, *Euler calculations for multielement airfoils using Cartesian grid*, AIAA Journal **24** (1986).

[28] W. J. Coirier and K. G. Powell, *An accuracy assessment of cartesian-mesh approaches for the Euler equation*, AIAA Paper **93-3335-cp** (1993), 423–437.

[29] ———, *A Cartesian, cell-based approach for adaptively-refined solutions of the Euler and Navier-Stokes equations*, 33rd Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 9 - 12 1995, AIAA-95-0566, pp. 1–14.

[30] P. Colella, *A direct Eulerial MUSCL scheme for fas dynamics*, SIAM J. Sci. Stat. Comput. **6** (1985), no. 1, 104–117.

[31] ———, *Multidimensional upwind methods for hyperbolic conservation laws*, Journal of Computational Physics **87** (1990), 171–200.

[32] P. Colella and P. R. Woodward, *The piecewise parabolic method (ppm) for gas-dynamical simulations*, Journal of Computational Physics **54** (1984), 174–201.

[33] W. R. Cowell and C. P. Thompson, *Tools to aid in discovering parallelism and localizing arithmetic in fortran programs*, Software – Practice and Experience **20** (1990), no. 1, 25 – 47.

[34] D. De Zeeuw and K. G. Powell, *An adaptively-refined Cartesian mesh solver for the Euler equations*, AIAA 10th Computational Fluid Dynamics Conference Proceedings, 1991, AIAA-91-1542-CR, pp. 166–180.

[35] ———, *Euler calculations of axisymmetric under-expanded jets by an adaptively-refined method*, 30th Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 6–9 1992, AIAA 92-0321, pp. 1–12.

[36] ———, *An adaptively refined Cartesian mesh solver for the Euler equations*, Journal of Computational Physics **104** (1993), 56–68.

[37] Darren L. De Zeeuw, *A quadtree-based adaptively-refined algorithm for solution of the Euler equations*, Ph.D. thesis, The university of Michigan, Michigan,USA, 1993.

[38] P. M. De Zeeuw, *Matrix-dependent prolongations and restrictions in a blackbox multigrid solver*, Journal of Computational and Applied Mathematics **33** (1990), 1–27, North-Holland.

[39] M. Van Dyke, *Perturbation methods in fluid mechanics*, The Parabolic Press, Stanford, California, 1975.

[40] ———, *An albnm of fluid motion*, The Parabolic Press, Stanford, California, 1982.

[41] P. R. Eiseman, *Coodinate generation with precise controls over mesh solver for the euler equations*, Journal of Computational Physics **47** (1982).

[42] ———, *Adaptive grid generation*, Computer Methods in Applied Mechanics and Engineering **64** (1987).

[43] B. Epstein, A. L. Luntz, and A. Nachshon, *Multigrid Euler solver about arbitrary aircraft configurations with Cartesian grids and local refinement*, AIAA 9th Computational Fluid Dynamics Conference, 1989.

[44] R. P. Fedorenko, *The speed of convergence of one iterative process*, USSR Comput. Math. and Math. Phys. **4** (1964), no. 3, 227–235, translation.

[45] H. Foerster and K. Witsch, *Multigrid software for the solution of elliptic problems on rectangular domains: Mgoo (release 1)*, Multigrid Methods (W. Hackbusch and U. Trottenberg, eds.), Lecture Notes in Mathematics, vol. 960, Springer, Berlin, 1982, Multigrid Methods Proceedings, pp. 427–460.

[46] P. O. Frederickson, *Fast approximate inversion of large elliptic systems*, Report 7 – 74, Lakehead University, Thunderbay, Canada, 1974.

[47] A. D. French, *Solution of the Euler equations on cartesian grids*, Ph.d. dissertation, College/of aeronautics, Cranfield University, Cranfield,Bedford,UK, 1991.

[48] S. K. Godunov, A. V. Zabrodin, and G. P. Prokopov, *A computational scheme for two-dimensional non stationary problems of gas dynamics and calculation of the flow from a shock wave approaching a stationary state*, USSR. Comput. Mathe. and Mathe. Physic. **1** (1962), no. 6, 1187–1219.

[49] W. Hackbusch, *On the convergence of multi-grid iterations*, Beit. Numer. Math. **9** (1978), 231 – 329.

[50] ———, *On the multi-grid method applied to difference equations*, Computing **20** (1978), 291 – 306.

[51] ———, *Survey of convergence proofs for multi-grid iterations*, Special Topics of Applied Mathematics (Bonn), North-Holland, Oct. 1979 1980, Proceedings, pp. 151 – 164.

[52] ———, *Multigrid convergence theory*, Multigrid Methods (W. Hackbusch and U. Trottenberg, eds.), Lecture Notes in Mathematics, vol. 960, Springer, Berlin, 1982, Multigrid Methods Proceedings, pp. 177–219.

[53] ———, *Parabolic multi-grid methods*, Computing Methods in Applied Science and Engineering VI (R. Glowinski and J. L. Lions, eds.), North-Holland, Amsterdam, 1984, Proc. 6th International Symposium, Versailles, Dec.1983, pp. 189 – 197.

[54] ———, *Multi-grid methods and applications*, Springer, Berlin, 1985.

[55] R. S. Herne-Smith, *Intelligent velocity profiler and data analysis for insertion flowmeters*, Ph.d. dissertation, School of Mechanical Engineering, Cranfield University, Cranfield,Bedford,UK, 1996.

[56] J. F. Dannenhoffer III, *A comparison of adaptive-grid redistribution and embedding for steady transonic flows*, AIAA 21st Fluid Dynamics, Plasma Dynamics and Lasers Conference (Seattle, WA), JUNE 18 – 20 1990, AIAA-90-1565, pp. 1 – 9.

[57] CFX International, *Cfx 4.2 manual*, AEA Technology Plc, Oxfordshire OX11 0RA, 1997.

[58] I. P. Jones and C. P. Thompson, *On the use of non-uniform grids in finite difference calculations*, Proceedings of BAIL I conference, Boole Press, 1980, pp. 332 – 341.

[59] C. W. Kitchens Jr., *Numerical experiments with the compressible Navier-Stokes equations*, Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics (H. Cabannes and R. Temam, eds.), vol. 1, Springer-Verlag, Berlin, July 3–7 1972, Lecture Notes in Physics, Vol. 18, pp. 120–129.

[60] S. L. Karman Jr., *SPLITFLOW: 3D unstructured cartesian/prismatic grid CFD code for complex geometries*, 33rd Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 9–12 1995, AIAA 95-0343, pp. 1–16.

[61] J. Krispin and H. M. Glaz, *Secon-order godunov methods and self-similar steady supersonic three-dimensional flowfields*, AIAA 22nd Fluid Dynamics, Plasma Dynamics and Laser Conference (honolulu, hawaii), JUNE 24–26 1991, AIAA 91–1653, pp. 1–14.

[62] A. M. Kuethe and C. Y. Chow, *Foundations of aerodynamics–bases of aerodynamic design*, fifth ed., John Wiley and Sons Inc., New York, 1998.

[63] Patrick A. Lezeau, *An adaptive quasi-newton coupled multigrid sover for the simulation of steady multiphase flows*, Ph.d. dissertation, School of Mechanical Engineering, Cranfield University, Cranfield,Bedford,UK, 1997.

[64] S. Li and L. Petzold, *Moving mesh methods with upwinding schemes for time-dependent PDEs*, Journal of Computational Physics **131** (1997), 368–377.

[65] M. Liou and Jr. C. J. Steffen, *A new flux splitting scheme*, Journal of Computational Physics **107** (1993), 23–39.

[66] C. Liu and Z. Liu, *High order finite difference and multigrid methods for spatially evolving instability in a planar channel*, Journal of Computational Physics **106** (1993), 92–100.

[67] ———, *Multigrid mapping and box relaxation for simulation of the whole process of flow transition in 3d boundary layers*, Journal of Computational Physics **119** (1995), 325–341.

[68] C. Liu, Z. Liu, and S. McCormick, *Multigrid methods for flow transition in three-dimensional boundary layers with surface roughness*, NASA Contractor Report 4540, NASA, September 1993.

[69] ———, *Multigrid methods for numerical simulation of laminar diffusion flames*, 31th Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 11–14 1993, AIAA 93-0236, pp. 1–11.

[70] F. Liu, *Multigrid solution of the Navier-Stokes equations with a two-equation turbulence model*, Frontiers of Computational Fluid Dynamics 1994 (D. A. Caughey and M. M. Hafez, eds.), John Wiley and Sons Ltd., Chichester, UK, 1994, pp. 339–359.

[71] F. Liu and S. Ji, *Unsteady flow calculations with a multigrid Navier-Stokes method*, 26th AIAA Fluid Dynamics Conference (San Diego,CA), JUNE 19–22 1995, AIAA 95-2205, pp. 1–10.

[72] B. Loyd and E. M Murman, *Fininte volume solution of the compressible boundary-layer equations*, NASA Contractor Report 4013, NASA, 1986.

[73] R. W. MacCormack, *Algorithmic trends in CFD in the 1990's for aerospace flow field calculations*, Algorithmic Trends in Computational Fluid Dynamics, Springer-Verlag Let., 1994, pp. 21–32.

[74] D. J. Mavripilis, *Multigrid techniques for unstructured meshes*, Computational Fluid Dynamics, Von Karman Institute for Fluid dynamics Lecture Series 1995–02, Von Karman Institute, March 13–17 1995, pp. 1–59.

[75] ———, *Unstructured mesh generation and adaptivity*, Computational Fluid Dynamics, Von Karman Institute for Fluid dynamics Lecture Series 1995–02, Von Karman Institute, March 13–17 1995, pp. 1–45.

[76] D. J. Mavriplis, *Accurate multigrid solution of the Euler equations on unstructured and adaptive meshes*, AIAA Journal **28** (1990), no. 2, 213–221.

[77] S. F. McCormick, *Multigrid methods*, Frontiers in Applied Mathematics, vol. 3, Society for Industrial and Applied Mathematics, Philadelphia,Pennsylvania, 1987.

[78] ———, *Multilevel adaptive methods for partial differential equations*, Frontiers in Applied Mathematics, vol. 6, Society for Industrial and Applied Mathematics, Philadelphia, 1989.

[79] J. E. Melton, M. J. Berger, M. J. Aftosmis, and M. D. Wong, *3D applications of a cartesian grid Euler method*, 33rd Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 9–12 1995, AIAA 95-0853, pp. 1–16.

[80] J. E. Melton, S. A. Pandya, and J. L. Steger, *3D Euler flow solutions using unstructured Cartesian and prismatic grids*, AIAA Paper **93-0331** (1993).

[81] J. M. Mendel, *Fuzzy logic systems for engineering: A tutorial*, Proceeding of the IEEE **83** (1995), no. 3, 345–377.

[82] D. S. Miller, *Internal flow systems*, BHRA fluid Engineering Series, vol. 4, BHRA Fluid Engineering, Cranfield, Bedford, 1978.

[83] C. R. Mitchell and R. W. Walters, *K-Exact reconstruction for the Navier stokes equations on arbitrary grids*, 31st Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 11–14 1993, AIAA 93-0536, pp. 1–16.

[84] R. A. Mitcheltree, M. D. Salas, and H. A. Hassan, *Grid embedding technique using cartesian grids for Euler equations*, AIAA Journal **26** (1988), 754 – 756.

[85] Y. J. Moon and H. C. Yee, *Numerical simulation by TVD scheme of complex shock reflections form airfoils at high angle of attack*, AIAA 25th Aerospace Sciences Meeting (Reno, NV), JANUARY 12–15 1987, AIAA 87-0350, pp. 1–17.

[86] Koji Morinishi, *A finite difference solution of the Euler equations on non-body-fitted Cartesian grids*, Computers Fluids **21** (1992), no. 3, 331–344.

[87] K. W. Morton and S. M. Stringer, *Finite volume methods for inviscid and viscous flows, steady and unsteady*, Computational Fluid Dynamics, Von Karman Institute for Fluid dynamics Lecture Series 1995–02, Von Karman Institute, March 13–17 1995, pp. 1–63.

[88] K. W. Morton and S.M. Stringer, *Finite volume methods for inviscid and viscous flows, steady and unsteady*, Computational Fluid Dynamics, Von Karman Institute for Fluid dynamics Lecture Series 1995–02, Von Karman Institute, March 13–17 1995, pp. 1–63.

[89] S. Murata, N. Satofuka, and T. Kushiyama, *Parabolic multigrid method for incompressble viscous flows using a group explicit relaxation scheme*, Comput. Fluids **19** (1991), 33 – 41.

[90] H. Paillere, K. G. Powell, and D. De Zeeuw, *A wave-model-based refinement criterion for adaptive-grid computation of compressible flows*, 30th Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 6–9 1992, AIAA 92-0322, pp. 1–10.

[91] L. W. Parker and R. G. Zalosh, *Godunov method and conputer program to determine the pressure and flow field associated with a sonic boom focus*, NASA Contractor Report NASA CR-2127, NASA, January 1973, Include Program Code for Godunov Method.

[92] S. V. Patankar, *Numerical heat transfer and fluid flow*, Hemisphere Publishing Corporation, Taylor and Francis Group, New York, 1980.

[93] S. V. Patankar, V. S. Pratap, and D. B. Spalding, *Prediction of laminar flow and heat transfer in helically coiled pipes*, Journal Of Fluid Mechanics **62** (1974), no. 3, 539 – 551.

[94] ———, *Prediction of turbulent flow in curved pipes*, Journal Of Fluid Mechanics **67** (1975), no. 3, 583 – 595.

[95] R. B. Pember, J. B. Bell, P. Colella, W. Y. Crutchfield, and M. L. Welcome, *An adaptive Cartesian method for unsteady compressible flow in irregular regions*, Journal Of Computational Physics **120** (1995), no. 2, 278 – 304.

[96] K. G. Powell, P. L. Roe, and J. Quirk, *Adaptive-mesh algorithms for computational fluid dynamics*, Algorithmic Trends in Computational Fluid Dynamics, Springer-Verlag Let., 1994, pp. 303–331.

[97] J. W. Purvis and J. E. Burkhalter, *Prediction of critical mach number for store configurations*, AIAA Journal **17** (1979), no. 11, 1170 – 1177, Article No. 79-4130.

[98] J. J. Quirk, *An adaptive grid algorithm for computational shock hydrodynamics*, Ph.d. dissertation, College of aeronautics, Cranfield University, Cranfield,Bedford,UK, 1991.

[99] ———, *An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies*, Computers Fluids **23** (1994), no. 1, 125–142, Also in ICASE Report 92 – 7.

[100] M. M. Rai, *A conservative treatment of zonal boundaries for Euler equation calculations*, Journal of Computational Physics **62** (1986), 472–503.

[101] S. S. Rao, K. Sundararaju, B. G. Prakash, and C. Balakrishna, *A fuzzy goal programming approach for structural optimization*, Tech. Report AIAA-92-2501-CP, AIAA, 1992.

[102] P. J. Roache, *Finite difference methods for the steady-state Navier-Stokes equations*, Proceedings of the Third International Conference on Numerical Methods

in Fluid Mechanics (H. Cabannes and R. Temam, eds.), vol. 1, Springer-Verlag, Berlin, July 3–7 1972, Lecture Notes in Physics, Vol. 18, pp. 138–145.

[103] U. Rüde, *Mathematical and computational techniques for multilevel adaptive methods*, Frontiers in Applied Mathematics, vol. 13, Society for Industrial and Applied Mathematics, Philadelphia,Pennsylvania, 1993.

[104] J. W. Ruge and K. stüben, *Algebraic multigrid*, Multigrid Methods,SIAM Frontiers in Applied Mathematics VOL.3 (S. F. McCormick, ed.), SIAM, Philadelphia, 1987, pp. 73 – 131.

[105] Rolf H. Sabersky, Allan J. Acosta, and Edward G. Hauptmann, *Fluid flow – a first course in flow mechanics*, third edition ed., Macmillan Publishing Company, New York, 1989.

[106] J. Saltzman, *An upsplit 3d upwind method for hyperbolic conservation laws*, Journal of Computational Physics **115** (1994), 153–168.

[107] Hermann Schlichting, *Boundary-layer theory*, seventh edition ed., McGraw-Hill Book Company, New York, 1979 (Translated by Dr. J. Kestin).

[108] M. J. Siclari and P. DelGuidice, *A multigrid finite volume method for solving the Euler and Navier-Stokes equations for high speed flows*, 27th Aerospace Sciences Meeting (Reno, Nevada), JANUARY 9 – 12 1989, AIAA-89-0283, pp. 1 –17.

[109] D. B. Spalding, *A novel finite difference formulation for differential expressions involving both first and second derivatives*, International J. for Numerical Methods in Engineering **4** (1972), 551–559.

[110] K. Stüben and U. Trottenberg, *Multigrid methods: Fundamental algorithms, model problem analysis and applications*, Multigrid Methods (W. Hackbusch and U. Trottenberg, eds.), Lecture Notes in Mathematics, vol. 960, Springer, Berlin, 1982, Multigrid Methods Proceedings, pp. 1–176.

[111] I. E. Sutherland and G.W. Hodgman, *Reentrant polygon clipping*, Communications of the ACM **17** (1974), no. 1, 32–42.

[112] C. P. Thompson, *The development and validation of a computational fluid dynamics code for global shared memory parallel architectures*, (1991), Submitted to Software–Practice and Experience.

[113] ———, *Developments in the solution of the incompressible Navier Stokes equations: Multigrid algorithms, parallelism and software tools*, Tech. Report BSC 91/5, IBM, Bergen Science Centre, Bergen, Norway, September 1991, D.Phil. dissertation.

[114] ———, *A parallel adaptive multigrid algorithm for the incompressible Navier-Stokes equations*, Asymptotic and Numerical Methods for Partial Differential Equations with Critical Parameters (H. G. Kaper and M. Garbey, eds.), NATO ASI Series, Kluwer Academic Publishers, 1993, pp. 293–309.

[115] C. P. Thompson, W. R. Cowell, and G. K. Leaf, *On the parallelization of an adaptive multigrid algorithm for a class of flow problems*, Parallel Computing **18** (1992), 449–466.

[116] C. P. Thompson, G. K. Leaf, and J. van Rosendale, *A dynamically-adaptive multigrid algorithm for the incompressible Navier-Stokes equations – validation and model problems*, Tech. Report MCS-103-0989, Argonne National Laboratory, 1989, (also in Applied Numerical Mathematics, Vol. 9, 1992).

[117] ———, *A dynamically-adaptive multigrid algorithm for the incompressible Navier-Stokes equations – validation and model problems*, Applied Numerical Mathematics 9 (1992), 511–532.

[118] C. P. Thompson, G. K. Leaf, and S. P. Vanka, *The application of a multigrid method to a buoyancy-induced flow problem*, Multigrid Methods (S. F. McCormick, ed.), Marcel Dekker, Dordrecht, 1988, pp. 605–629.

[119] M. C. Thompson and J. H. Ferziger, *An adaptive multigrid technique for the incompressible Navier-Stokes equations*, Journal of Computational Physics **82** (1989), no. 1, 94–121.

[120] H. S. Udaykumar, H. Kan, W. Shyy, and R. Tran-Son-Tay, *Multiphase dynamics in arbitrary geometries on fixed Cartesian grids*, Journal of Computational Physics **137** (1997), 306–405.

[121] S. P. Vanka, *Block-implicit multigrid solution of Navier-Stokes equations in primitive variables*, Journal of Computational Physics **65** (1986), 138–158.

[122] J. C. T. Wang and G. F. Widhopf, *A high-resolution TVD finite volume scheme for the Euler equations in conservation form*, AIAA 25th Aerospace Sciences Meeting (Reno, NV), JANUARY 12–15 1987, AIAA 87-0538, pp. 1–17.

[123] ———, *Numerical simulation of blast flowfields using a high resolution TVD finite volume scheme*, Computers and Fluids **18** (1990), no. 1, 103–137.

[124] Z. J. Wang, *A fully conservative structured/unstructured chimera grid scheme*, 33rd Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 9 – 12 1995, AIAA-95-0671, pp. 1 –10.

[125] Z. J. Wang and H. Q. Yang, *A unified conservative zonal interface treatment for arbitrarily patched and overlapped grid*, 32nd Aerospace Sciences Meeting and Exhibit (Reno, NV), JANUARY 10 – 13 1994, AIAA-94-0320, pp. 1 –10.

[126] Z. U. A. Warsi, *Fluid dynamics – theoretical and computational approaches*, CRC Press, Boca Raton, Florida, 1993.

[127] B. Wedan and J. C. South Jr, *A method for solving the transonic full-potential equation for general configurations*, AIAA Paper **83-1869** (1983).

[128] P. Wesseling, *Numerical solution of the stationary Navier-Stokes equations by means of a multiple grid method and Newton iteration*, Tech. Report Report NA – 18, Delft University of Technology, 1977.

[129] ———, *The rate of convergence of a multiple grid method*, Lecture Notes in Mathematics (G. A. Watson, ed.), vol. 773, Springer, Berlin, 1980, Numerical Analysis. Proceedings, Dundee 1979, pp. 164 – 184.

[130] Pieter Wesseling, *An introduction to multigrid methods*, John Wiley and Sons Ltd., Baffins Lane, Chichester, England, 1992.

[131] H. Weyl, *Concerning the differential equations of some boundary layer problems*, Pro. Natl. Acad. Sci. **27** (1941), 578 – 583.

[132] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, Journal of Computational Physics **54** (1984), 115–173.

[133] G. yang, *A Cartesian cut-cell method for axisymmetric separating body flows*, 27th AIAA Fluid Dynamics Conference (New Orleans, LA), JUNE 17–20 1996, AIAA-96-1973, pp. 1 –10.

[134] G. Yang, D. M. Causon, D. M. Ingram, R. Saunders, and P. Batten, *A Cartesian cut cell method for compressible flows part a: Static body problems*, The Aeronautical Journal (1997), 47–56.

[135] ———, *A Cartesian cut cell method for compressible flows part b: Moving body problems*, The Aeronautical Journal (1997), 57–65.

[136] J. Y. Yang, J. C. Huang, and C. A. Hsu, *A high-order streamline godunov scheme for steady hypersonic equilibrium floes*, AIAA 24th Fluid Dynamics Conference (Orlando, FL), JULY 6 – 9 1993, AIAA 93–2997, pp. 1–10.

[137] L. S. Yao and S. A. Berger, *Entry flow in a curved pipe*, Journal Of Fluid Mechanics **67** (1975), no. 1, 177 – 196.

[138] H. C. Yee, R. F. Warming, and A. Harten, *Implicit total variation diminishing (DVT) schemes for steady-state calculations*, Journal of Computational Physics **57** (1985), 327–360.

[139] S. Yoon and D. Kwak, *Multigrid convergence of an lu scheme*, Frontiers of Computational Fluid Dynamics 1994 (D. A. Caughey and M. M. Hafez, eds.), John Wiley and Sons Ltd., Chichester, UK, 1994, pp. 319–338.